basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

IMPLEMENTATION PLAN FOR STANDARDISING PROGRAMMING LANGUAGE FOR INFORMATION TECHNOLOGY FOR 2015

1. PROBLEM STATEMENT

The complexity to synchronise two different programming languages to implement the Information Technology (IT) curriculum has been evident since the implementation of the National Curriculum Statement (NCS) in 2006 and the National Senior Certificate (NSC) in 2008, and has become even more evident since the implementation of the IT Curriculum and Assessment Policy Statement (CAPS).

Also, the implementation of the IT CAPS in Grade 11 in 2013, pointed out several complexities regarding Java implementation of especially the database content in the CAPS. These complexities are not intended by the IT CAPS content and add additional overhead for Java learners which could eventually impact on curriculum delivery. It became clear, that Java is not the best language to implement for the IT CAPS curriculum due to these intrinsic complexities of the Java language.

Implementing the IT curriculum using two programming languages impacts in terms of several factors, including different levels of complexity dealt with regarding the different content areas in the IT curriculum (See **Annexure B**).

Factors:

1.1. Curriculum

With Java there is an additional layer of complexity between the concepts and the implementation (which solidifies a pupil's grasp of the concept). The complexity of translating abstract concepts into practical implementation is much reduced with Delphi syntax, inherently stable runtime environment and the simplicity of dependencies, reduced complexity and speed of compilation. Even though Delphi is much less suited to complex, large scale highly integrated enterprise solutions it is much better suited to classroom code based representation of generic abstract programming concepts.

It is a well-known fact that the programming language used to implement a curriculum, should be best suited to implement the specific curriculum and also consider the target audience.

Literature also suggests that Java is not a suitable language for introductory programming courses (See **Annexures B and C**)

"a computing curriculum should not become a vocational training ground for current industrial-strength programming languages and programming tools. Any introductory course in Computing should not be arranged around the syntax of a currently fashionable programming language. It is more important to concentrate on principles. At the same time, however, a curriculum must also teach how these principles apply to the real world, but this relates to teaching program design principles not the use of language constructs. Teach good habits early otherwise bad habits become ingrained and require costly fixes. To avoid any confusion, the course should not use a complex language that distracts from design principles and should not pose problems from complex application domains. The first language should facilitate the teaching of design principles. " The Structure and Interpretation of the Computer Science Curriculum, Matthias Felleisen et al, Journal of Functional Programming(2004), 14: 365-378 Cambridge University Press.

Java, C, C++, C# fall into the category of complex programming languages. The design principles that should be covered are data abstraction, functional/procedural abstraction, data-directed programming.

1.2. One national examination:

1.2.1. Exam panels

Not standardising using one programming language suggests that chief examiners and moderators ideally need to know different programming languages which include the syntax, IDE (Integrated Development Area), databases used, database connections, query languages, etc. or that the panels should be carefully balanced between languages.

1.2.2. Question setting

The different approaches in different languages complicate question setting as some content such as database manipulation is less complex in some languages than in others. At times, one could argue that such issues advantage/disadvantage candidates in one specific language, as they have to deal with more overheads, code more lines, deal with more complex concepts, etc. This could impact on the fairness of assessment.

1.2.3. Possible effect on marking

The choice of a programming language is not regulated by policy and potentially schools could choose a language according to the definition in the CAPS, which could result in provinces having to deal with different languages (see 2.5). Therefore, not standardising implies that markers would need to know different programming languages and marking could become complicated.

1.3. Migration of learners

It is problematic for learners moving between provinces/schools that use different programming languages and often such a learner has no choice but to drop the subject.

1.4. Migration of teachers

Teachers moving between provinces/schools that use different programming languages need training and support to master the differences in syntax, approach, etc.

1.5. Teacher support

IT teachers are scarce and subject support is specialised. With more than one programming language, the support is split. Instead of building strong support and resources in one language, these are split and teachers sometimes struggle to find enough support and resources or to share resources across provinces.

1.6. Teacher training

Teacher training programmes either need to make provision for more than one high level language (which is problematic in terms of time available, depth, etc.) or only train teachers for one specific language (with the result that teachers only feel comfortable to teach in specific provinces/schools).

1.7. Technical issues

Various setups, approaches, versions of Java/Delphi as well as Netbeans, databases, database connection, query language, etc. need to be considered. Also, using different languages, Integrated Development Environments (IDEs), databases, query languages, etc. as well as different versions of programming languages, IDEs, databases, etc., raise complexity and call for error in national practical examinations.

For Java, the correct combination of versions for Java JDK, Netbeans, and JavaDB need to be used to avoid problems.

Delphi is syntax and environment stable – Java is not – Learning runtime environment dependencies and configuration (JRE's, Class paths, Class version dependencies and conflicts, Java container's and their individual features, characteristics module loading techniques etc.) goes far beyond what any secondary school pupil should be exposed – Delphi is a static and stable language and runtime and these concerns are for the most part negated.

1.8. Other support

The subject has a small number of learners, teachers and subject advisors. If support and training have to focus on different programming languages to implement the curriculum, it becomes time consuming, impractical and costly. Also, developing material needs to be done in two programming languages, one of the possible reasons for IT not having Grade 11 and 12 CAPS textbooks listed on the national catalogue.

2. BACKGROUND

2.1. Information Technology was introduced in Grade 10 in 2006 with the implementation of the NCS and was developed from Computer Studies HG offered in Report 550. With Report 550, provinces wrote their own provincial papers and therefore could choose their own programming language. Two programming languages, Java and Delphi, were offered by the different provinces respectively, where 5 provinces (EC, FS, GP, LP, NW) offered Delphi and 4 provinces (KZN, MP, NC, WC) offered Java. This status quo remained when Information Technology was introduced in Grade 10 in 2006 and the NSC examinations currently provides for both Delphi and Java.

2.2. Several problems were recognised due to having two programming languages, including deriving detailed content specification considering the different approaches and peculiarities of each as well as setting Grade 12 NSC papers for one national examination, considering all of these and avoid disadvantaging any group of learners. Theoretically, it should not be a problem but practice suggests otherwise.

2.3. In 2010/11, the DBE proposed a switch to one programming language and after investigation recommended Delphi/Object Pascal. However, after feedback from Java provinces, the DBE suggested that:

2.3.1. The status quo (a choice between Java and Delphi) is retained in terms of the high-level programming language and

2.3.2. Provinces/schools that will continue using Java will have to standardise using Netbeans (GUI Builder as required by CAPS) with Java, though the DBE still strongly recommended Delphi for implementing the IT CAPS.

2.4. The proposal was again tabled at HEDCOM in March 2013 and HEDCOM gave instructions to prepare an implementation plan to change to one programming language (Delphi) to implement the IT curriculum.

2.5. In terms of policy, the IT CAPS does not mention any specific programming language, but only describes criteria for the programming language to be used:

A high-level software development tool that includes an integrated development environment which:

- *supports both structured and object oriented methodologies*
- *uses a visual development environment with a graphical user interface builder and*
- *allows for event-driven programming*

The GUI builder should allow for component based development with a WYSIWYG (what you see is what you get) editor utilising an event driven architecture.

The criteria describe a Rapid Application Development (RAD) tool. Two of the most popular RAD systems for Windows are Visual Basic and Delphi, though Delphi is regarded the world's best RAD tool.

The fact that the CAPS does not mention any specific language, means that the choice of a particular programming language is not regulated by policy and a school could opt for any programming language that satisfies the requirements. Currently the NSC examination provides for two programming languages, namely Delphi and Java. It is therefore safe to derive that a school is currently free to choose between these two languages, however it could be argued that any other high level language that meets the criteria could be used, should the DBE not formally regulate the language.

2.6. With the implementation of the CAPS it became clear that, though Java can implement the IT CAPS, some aspects in the curriculum, e.g. databases are more complex to implement in Java and requires using concepts outside the IT CAPS curriculum, such as using record lists as containers when dealing with the database and including the Java Persistence Query Language which requires queries over objects. (See **Annexure B** for a comparison between Delphi and Java with regard to the IT CAPS requirements).

2.7. One should also note, that after the Grade 12 CAPS training in February 2013, IT teachers from Mpumalanga that attended the training, initiated a change from Java to Delphi, after realising the complexity of Java over Delphi. This was supported by the majority of IT teachers in the province.

Also, talking to some of the teachers, one commented: "the only thing I regret, is not having changed to Delphi long ago".

Also, during the training of KZN subject advisors and teachers in April 2013, some teachers expressed their concerns about database aspects in Java and a few expressed considering changing to Delphi.

2.8. The choice of a suitable programming language in schools is a contentious issue that normally raises a lot of emotion and would probably be contested as teachers would approach this from their own contexts, backgrounds and experiences as well as the contexts of their schools and learners.

3. DISCUSSION

3.1. One should note that the objective of a subject at school level is to provide a broad foundation that feeds into various disciplines in the particular field in which learners could specialise after school receiving further training at a higher education institution, as well as to develop critical thinking. The aim of IT at school includes motivating learners to be enthusiastic about IT and to realise the importance of the subject in terms of critical thinking, to learn to think clearly and logically, to pay attention to detail and accuracy, and to instil perseverance (keep trying), resilience and self-confidence as well as to realise the value of IT in terms of future careers. The aim is **not** to produce professional programmers without further education (just as Life Science do not produce nurses or doctors after Grade 12).

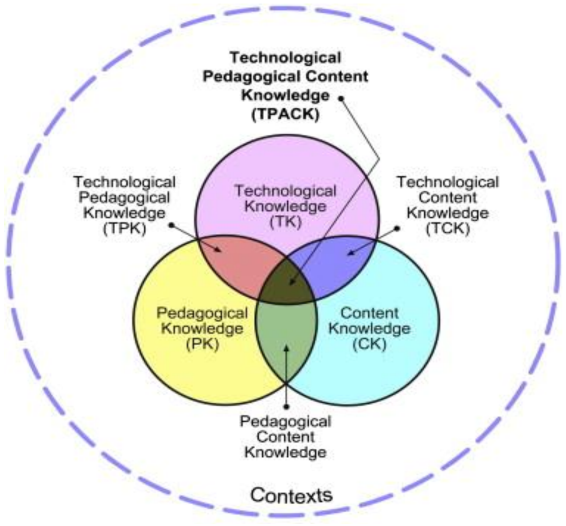
3.2. Literature suggests that programming is hard to learn and that the choice of the environment and tool used for teaching is therefore very important:

"Programming is a skill that is considered hard to learn and even after two years of instruction, the level of programming understanding is low." (Kurland et al., 1989).

"Learning to program is generally considered hard, and programming courses often have high dropout rates. It has even been said, that it takes about 10 years for a novice to become an expert programmer." (Soloway, E. & Spohrer, J. (1989)).

"However, if supported by suitable teaching strategies and tools it can be mastered by pupils to some extent." (Papert, 1980).

3.3. In addition to the above, one needs to note that with IT, as compared to other subjects, teachers need not only have to cope with subject content knowledge and subject pedagogy knowledge, but also with technology knowledge as well as the overlap and integration of these, as illustrated in the following diagram:


Technological Pedagogical Content Knowledge (TPACK)
Technological Pedagogical Knowledge (TPK)
Technological Knowledge (TK)
Technological Content Knowledge (TCK)
Pedagogical Knowledge (PK)
Content Knowledge (CK)
Pedagogical Content Knowledge
Contexts

Likewise, learners need to cope with both subject content knowledge as well as technology knowledge, i.e. the programming environment.

When learning to program learners need to cope with, e.g.,

- learning the language features
- program design
- program comprehension

3.4. Analysis of NSC results over the past 5 years suggests that the subject is declining in terms of numbers.

The number of schools offering IT in from 2008 to 2012:

National	#Schools	Increase/Decrease
2012	359	6
2011	353	-28
2010	381	-44
2009	421	-14
2008	439	

TABLE 1: THE NUMBER OF SCHOOLS OFFERING IT FROM 2008 – 2012

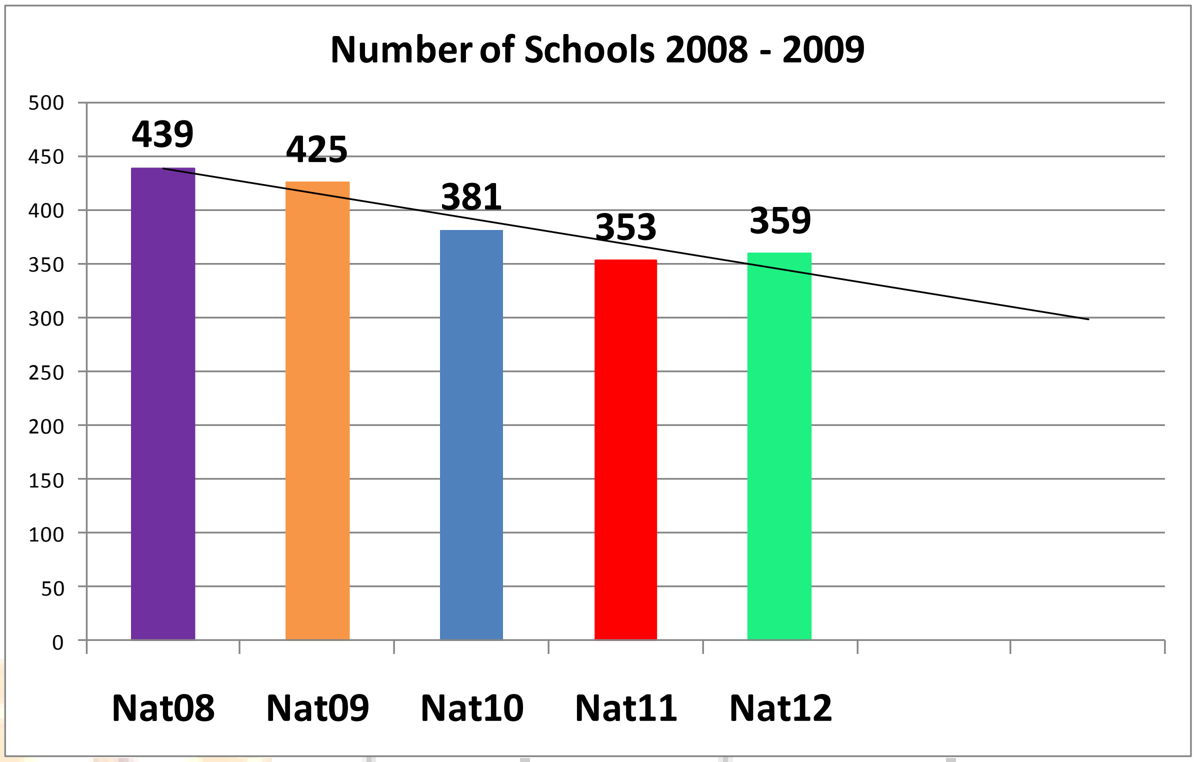


FIGURE 1 NUMBER OF SCHOOLS 2008 – 2012

The number of schools offering IT has declined by about 20% since 2008 up to 2011 and shows a very slight increase again in 2012. The increase, however, could be due to provinces 'inheriting' centres from ERCO.

The number of learners offering IT from 2008 – 2012:

National	#Learners	Increase/Decrease
2012	4428	115
2011	4313	-571
2010	4884	-1362
2009	6246	-541
2008	6787	

TABLE 2: THE NUMBER OF LEARNERS OFFERING IT FROM 2008 – 2012

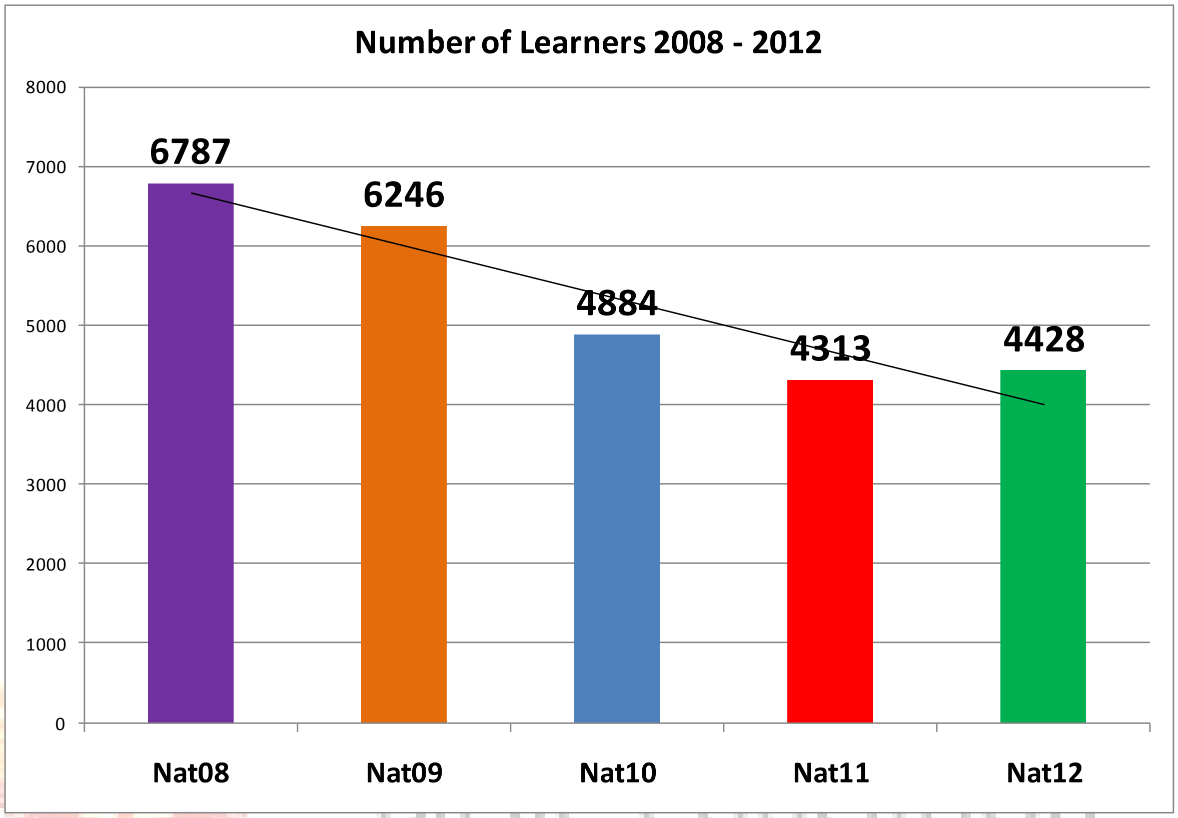


FIGURE 2 NUMBER OF LEARNERS FROM 2008 – 2012

The numbers show a decline of about 36% in learner numbers from 2008 to 2011, with a slight increase again in 2012. As mentioned above, it could be due to 'inheriting' schools from ERCO.

3.5. Factors to take into account when considering a programming tool to implement the IT curriculum should be based on literature, the IT CAPS, sound pedagogy and criteria suggested by literature. (See **Annexure C**)

3.5.1. Milbrandt suggests that a programming language to be used in education should be

- easy to learn
- structured in design
- universal in use and
- powerful in computing capacity

3.5.2. In the paper, Teaching Mathematics and Programming – New Approaches with Empirical Evaluation (p11 – 19); Linda Mannila adds that the language should also have

- a simple syntax (intuitive, easy to read)
- use meaningful keywords
- provide easy I/O handling and output formatting.

3.6. As several papers state that “*Learning to program is generally considered hard, and programming courses often have high dropout rates*”, it is important that the environment/tool that is used for learning programming is simple and streamlined.

3.7. Literature also suggests the following important aspects to be considered when choosing a programming language:

- the curriculum to be implemented
- the target audience

3.8. Should the IT CAPS curriculum be considered, the following curriculum aspects need to be taken into account:

- Event-driven programming
- Object orientated paradigm
- Manipulation of a database through code constructs

One should therefore consider a programming language that handles all the above aspects with ease.

3.9. Furthermore, it should also be noted that the Assessment and Qualifications Alliance's (AQA) in the UK announced in 2010 that A-level computer science students will no longer be taught C, C# or PHP from next year (2011). The following is an extract from the statement in The Register:

The board "*highly recommended" switching to Pascal/Delphi*[1] *because it is stable and was designed to teach programming and problem solving. Teachers planning to use Java are warned that many universities are considering dropping it from their first year computer science programmes, "as has happened in the US*".

3.9.1. The document detailing the withdrawal, states: (See **Annexure D**)

"Pascal/Delphi is highly recommended because it was designed specifically to teach programming and problem solving - see http://uva.onlinejudge.org/ - and it is stable. Its event-driven forms-based object Pascal manifestation, Delphi, has excellent support for a range of applications from networking through graphics to databases. Delphi is still rated as the world's best RAD system and is used extensively throughout the world for commercial application development".

3.9.2. The document also states

.."a computing curriculum should not become a vocational training ground for current industrial-strength programming languages and programming tools."

Also

"To avoid any confusion, the course should not use a complex language that distracts from design principles and should not pose problems from complex application domains."

4. RECOMMENDATION

4.1. In light of the above, considering the current two languages used, DBE recommends that all provinces standardise using Delphi as programming language.

Delphi is ideal for learning programming as it is a strict, yet forgiving language. You don't need to worry about things such as case, its compiler tells you where your errors are, it offers console mode, desktop mode, the ability to learn all the basic constructs up to advanced multi-tier technologies, all with the same language - something that is VERY important learning about today's computing

[1] [1] (Object Pascal) is the object oriented descendant of Pascal

world. It is also easy for teachers to get up and running and there are excellent resources.

It is a mature language. Delphi is a stable with a proven track record.

New versions of Delphi offer Mac, iOS and Android compilers from the same IDE, and same language! This gives a GREAT foundation for teaching students about the differences between platforms in a simple way as they don't need to learn a lot of new things to be able to make it accessible.

4.2. The recommended date of implementation is 2015 for Grade 11 and 2016 for Grade 12 with teacher training in 2014 and 2015 for Grade 11 and Grade 12 respectively (see point 7 – Implementation)

4.3. It should be noted that the recommendation is made in light of the current two programming languages used to implement the IT curriculum.

5. IMPLICATIONS FOR IMPLEMENTING DELPHI AS PROGRAMMING LANGUAGE ACROSS PROVINCES

5.1. BENEFITS

5.1.1. Resources

i. Availability of resources – Resources are available and no need to develop resources from scratch.

  a. Textbook

  The national catalogue lists no textbooks for IT for Grades 11 and 12, however a CAPS-compliant textbook for Delphi is being developed (needs to be evaluated and approved by DBE)

  b. Other resources

  Several websites and blogs are available that support Delphi. Embarcadero (owners of Delphi) and EOH (distributers of Delphi in Africa) are willing to support IT in schools through workshops, material, etc. (See **Annexure C**)

ii. Sharing of resources between and across provinces – learners and teachers can use resources such as assessment resources from all provinces

iii. No need to develop all resources for two languages – saves time and money.

5.1.2. Cooperation between and across provinces

Improved cooperation between and across provinces. No language 'fights' and tension between provinces anymore, no longer 'us' and 'them' or 'Java' and 'Delphi' – united 'in one language' working for the benefit of the subject and learners in our country.

5.1.3. Examination papers set for only one language (Delphi)

i. Time to set and moderate practical papers will reduce, resulting in cost saving

ii. Question papers that are fair to all learners – no disadvantaging learners from a specific programming language

iii. No need to 'balance' examination panels in terms of programming language

iv. Chief examiners and moderators are not ideally required to know both programming languages

v. Provinces outsourcing the marking of IT papers can outsource to any province

5.1.4. Curriculum support

Better, 'undivided' support to all provinces – support focusing on one language only, possibly saving time and money.

5.1.5. Technical issues

Focus on one set of requirements, reducing possibility for error. Also, Delphi has an integrated IDE and is backwards compatible, reducing the risk of incompatibility of different versions.

5.1.6. Teacher training and development

HEIs training teachers in one high-level programming language focusing on content depth and better preparing teachers for the classroom

5.1.7. Migration issues for learners and teachers are solved.

5.1.8. Curriculum implementation

Considering the current situation, Delphi is the best language to implement the IT CAPS curriculum and learners will not be required to learn concepts and deal with complexities that is not intended by the IT CAPS curriculum

5.1.9. Other

New versions of Delphi have available Mac, iOS and Android compilers, all from the same IDE and same language! This gives a GREAT foundation for learners to learn about the differences between platforms in a simple way as they don't need to learn lots of new things to be able to make it accessible.

5.2. CHALLENGES

5.2.1. Resistance from Java provinces (WC, KZN, NC – majority of MP teachers already supported the change to Delphi)

- Java provinces may see the introduction of Delphi as the common programming language across provinces as a 'win-loose' situation where they are the 'losers'
- Ego's of individuals who were part of the initial decision to use Java to implement the curriculum (however, this should NOT be a reason not to implement Delphi – sound educational considerations should be considered) and attitudes of 'anything but Delphi'

5.2.2. Teachers are 'change weary'

Teachers from Java provinces may be reluctant to change or question the fact the 'they' have to change whilst 'those' from other provinces do not have to.

5.2.3. Teacher training required in Java provinces

Teachers from Java provinces will need Delphi training. However, one needs to note that the concepts are the same and training will only need to focus on the 'new environment'. Also, teachers that have been in the system for more than 10 years, taught Pascal in the past and should adapt very easily to Delphi. Also considering that a textbook is available, a 2-day training for Grade 11 and Grade 12 each, should suffice. (See **Annexure A** for cost analysis)

5.2.4. Argument for industry relevance/other programming languages

The aim of school is not to provide vocational training (that is the task of higher education institutions or FET colleges) but to lay a solid foundation to enable a learner to pursue further education at an HEI in the IT field.

Although Java is more widely used in industry, Delphi is a matured language (unlike other 'newer' languages that may still struggle with developmental issues), comes from Pascal (originally designed to teach programming), used by about 2 million developers across the world and provides a stable environment.

5.3. COST IMPLICATIONS

Costs involved implementing a new programming language generally includes:

- Teacher training
- Software licenses
- Development of resources

5.3.1. Teacher training

Teachers in Java provinces will require training in Delphi. However, Mr Buitendag, IT lecturer from Tshwane University of Technology (TUT) is willing to train subject advisors and teachers at no cost (except for travelling and accommodation expenses) providing the training takes place during holidays. See **Annexure A** for estimated cost implications.

5.3.2. Software licenses

Embarcadero offers free licences to Delphi schools and learners for a limited period, i.e. they need to register annually to benefit from this offer. (See **Annexure B**)

5.3.3. Development of resources

Delphi resources are available, including a CAPS compliant textbook for Grades 11 and 12, though it has not been evaluated and catalogued.

6. IMPLEMENTATION PLAN

6.1. The implementation plan considers teacher training, resources required, e.g. software licenses, support material and timeframes

6.2. In order to effectively implement the standardisation, suggested activities and timeframes are provided in the table below:

Activity	Time Frame	Responsibility
Communicate standardisation of programming language (Delphi) for IT (Circular to provinces)	**September 2013**	DBE (Curriculum)
Develop teacher training material for Grade 11 CAPS content	**February 2014**	DBE (Curriculum) and PDEs
Procuring of Delphi licences for provinces	**March 2014**	DBE (Curriculum) and PDEs
Teacher training – Grade 11 CAPS content	**April and/or June 2014**	DBE (Curriculum) and PDEs
Implementation of Delphi in all provinces for Grade 11	**January 2015**	PDEs
Develop teacher training material for Grade 11 CAPS content	**February 2015**	DBE (Curriculum) and PDEs
Teacher training – Grade 12 CAPS content	**April and/or June 2015**	DBE (Curriculum) and PDEs
Teacher training – Grade 11 CAPS content	**April and/or June 2014**	DBE (Curriculum) and PDEs
Implementation of Delphi in all provinces for Grade 12	**January 2016**	PDEs

6.3. Risk management

Risk description	**Risk Management**
Inadequate budgets for planned activities	Budgets made available for teacher training
Ineffective support from districts	DBE capacitate subject advisors
Inadequate monitoring and supporting	Monitor and support processes and systems put in place

6.4. Conditions for effective implementation

Effective implementation would include the following conditions:

- Java provinces buy into the standardisation of the programming tool and Delphi as the recommended tool
- Training for subject advisors and teachers from Java provinces
- DBE Support for teachers and subject advisors
- Availability of textbooks for Grade 11 and Grade 12
- Excellent, additional support material, e.g. video's, exemplar assessment tasks available

7. CONCLUSION

The Implementation Plan is aligned to the support that will be required for standardising the tools across the country. It will also provide an opportunity to consolidate and strengthen IT as a subject.

Annexure A – Analysis of cost

Due to the fact that the concepts in all programming languages are the same, a 2-day training workshop each for Grade 11 and Grade 12 is recommended.

1. Two-day teacher training for Grade 11 content (2014/15 financial year)
    a. Suggested focus for the workshop:
        - Delphi IDE (GUI)
        - Delphi syntax
        - Delphi and Database

    Cost for the workshop involves 1 night's accommodation for teachers, meals and travel expenses as well as travel and accommodation for the presenter.
    The number of teachers per province is estimated from the number of schools offering IT in the province as per NSC 2012 results.

	KZN	**MP**[2]	**NC**	**WC**
Number of	**92**	**18**	**8**	**60**
1 night @ R1500.00 pp (incl. breakfast and dinner)	R138 000.00	R27 000.00	R12 000.00	R90 000.00
Lunch @ R150.00 pp x 2	R27 600.00	R5 400.00	R2 400.00	R18 000.00
Travel @ average 300 km pp @ current fuel rate (R3.90 per km)	R107 640.00	R21 060.00	R9 360.00	R70 200.00
Travel and accommodation for presenter from TUT (plane ticket + 1 night accommodation)	R7 000	R7 000	R7 000	R7 000
Printing cost – training material	R3 000.00	R3 000.00	R3 000.00	R3 000.00
Total	±R285 000.00	±R65 000.00	±R35 000.00	±R90 000.00

[2] MP teachers already had 1 day training in April 2013 (at their own cost, offered by Mr Buitendag from TUT) and may require less training

2. Two-day teacher training for Grade 12 content (2015/16 financial year)
    a. Suggested focus for the workshop:
        - Delphi and Database
        - Own class

    Cost for the workshop involves 1 night's accommodation for teachers, meals and travel expenses as well as travel and accommodation for the presenter.
    The number of teachers per province is estimated from the number of schools offering IT in the province as per NSC 2012 results.

	KZN	**MP**[3]	**NC**	**WC**
Number of	**92**	**18**	**8**	**60**
1 night @ R1500.00 pp (incl. breakfast and dinner)	R138 000.00	R27 000.00	R12 000.00	R90 000.00
Lunch @ R150.00 pp x 2	R27 600.00	R5 400.00	R2 400.00	R18 000.00
Travel @ average 300 km pp @ current fuel rate (R3.90 per km)	R107 640.00	R21 060.00	R9 360.00	R70 200.00
Travel and accommodation for presenter from TUT (plane ticket + 1 night accommodation)	R7 000	R7 000	R7 000	R7 000
Printing cost – training material	R3 000.00	R3 000.00	R3 000.00	R3 000.00
Total	**±R285 000.00**	**±R65 000.00**	**±R35 000.00**	**±R90 000.00**

[3] MP teachers already had 1 day training in April 2013 (at their own cost, offered by Mr Buitendag from TUT) and may require less training or no training at all

ANNEXURE B

IT CAPS requirements and DBE investigation

The prominent requirements for the selection of a suitable programming language for the CAPS document, is stipulated on page 10 of the CAPS document;

Requirements for high-level programming tool to be used for software development:

- High-level software development tool that includes an integrated development environment (IDE) which:
- supports both structured and object oriented methodologies
- uses a visual development environment with a graphical user interface builder
- allows for event driven programming

The GUI builder should allow for component based development with a WYSIWIG (what you see is what you get) editor utilising an event driven architecture.

The criteria describe a RAD (rapid application development) tool. Two of the most popular RAD systems for Windows are Visual Basic and Delphi, though Delphi is regarded as the world's best RAD tool

	Criteria **CAPS**	**Visual Basic.NET (Microsoft Visual Studio)**	**Delphi (Embarcadero RAD Studio)**	**Java (NetBeans)**
1	*High-level software development tool that includes an integrated development environment (IDE)*	Yes	Yes	Yes
2	*supports both structured* *(Structured Programming)*	Yes	Yes	No (Not without utilising techniques to bend/side-step the intended purpose of the language)
3	*object oriented methodologies*	Yes	Yes	Yes
4	*uses a visual development environment with a graphical user interface builder*	Yes	Yes	Yes
5	*allows for event driven programming See -* ***[TeachNote *1]***	Yes	Yes	Yes

6	*allow for component based development with a WYSIWIG (what you see is what you get) editor*	Yes	Yes	Yes
7	*Editor utilising an event driven architecture.*	Yes	Yes	Yes
8	*The development tool could also include software design utilities to facilitate the application of software engineering practices.*	Yes	Yes	Yes

Literature suggestions

In an article written by Siegfried, Chays and Herbert (2008) entitled: *Will There Ever Be Consensus on CS1?* a cognitive comparison table based on six of McIver's rules for evaluating an introductory programming language is listed. The six criteria are listed below:

1) "Closeness of mapping" addresses how well the notation represents the domain for which it is intended, e. g., if we are trying to describe arithmetic, how closely does our notation resemble arithmetic?
2) To be "consistent", similar semantics should be expressed in similar syntax. Therefore, an if...elseif...else construction would be considered more consistent than a switch statement.
3) "Diffuseness" refers to the verbosity of the language. COBOL would be an example of a diffuse notation.
4) "Error-prone" constructions are those that are more likely to lead to errors, or perhaps even encourage them. The use of separate pairs of brackets for different dimensions of an array might be considered error-prone.
5) "Hard mental operations" would require the programmer to prefer potentially difficult tasks in writing a program, e.g., entering all numeric constants in an unusual number base.
6) "Role expressiveness" refers to the ability of a reader to infer the usage of a feature just from its structure.

They presented a table which represents the cognitive comparison of various programming languages compared. Using the same approach the following table aims to compare the three programming languages as presented in this document. Two of which i.e. Java & Pascal was also part of the study done by Siegfried, et.al

Dimension	***Optimal***	**Java**	**Pascal (Delphi)**	**VB.NET**
Closeness of Mapping	***High***	Low	Medium	Medium
Consistency	***High***	Low to Medium	Low to Medium	Low to Medium
Diffuseness	***Medium to High***	Low	Low to Medium	Low to Medium
Error proneness	***Low***	Medium to High	Low to Medium	Medium (VB6 vs. .Net)
Hard Mental Operations	***Low***	Medium to High	Low to Medium	Low to Medium
Role Expressiveness	***High***	Low	Medium to High	Medium (Need to utilize .Net constructs e.g. strings & DB's)

The CAPS document also includes the development of Data aware applications (Grade 11 p 32 – p 33) where a connection to a database needs to be established to apply transactions.

The following table lists the various objectives and contrast the technical level required for implementing the objective in each of the three programming environments. Brief notes on each of the objectives are also supplied.

The complexity scale used for indication purposes are as follows:

1) Simple [Very easily accomplished]
2) Relatively simple [Some initial background/base knowledge required]
3) Somewhat technical [technical requiring some detailed knowledge which is accomplished with “limited” / modest technical code constructs]
4) Technical [technical requiring more detailed knowledge which is accomplished with more code constructs]
5) Complex process [very technical with a very detailed knowledge required with expanded code constructs]

For illustration/comparison purposes connection to an ADO database e.g. MS Access 2007 database is assumed.

Objective	**Visual Basic.NET (Microsoft Visual Studio)**	**Delphi/Pascal (Embarcadero RAD Studio)**	**Java (NetBeans)** **ADO database e.g. MS Access 2007 database and SQL assumed**	**Java (NetBeans)** **Suggested for CAPS by Java advisors** (JavaDB and JPQL)**
Accessing a database through programming language constructs & *Setup a connection or connect to a database (single table) by providing path in code statements*	**Complexity Scale: 2 – 3** Using a connection string and command a data adapter with a data table or dataset	**Complexity Scale: 2 – 3** Using a connection string and a ADO Connection component as well as a Data Source and query or table component	**Complexity Scale: 3** Using a connection string and a connection object and a result set object. Knowledge of exceptions is required in the establishment of a connection object	**Complexity Scale: 4-5** In order to create and establish a DB and a subsequent connection to the DB more background knowledge will be required, especially to concepts such as client server computing where knowledge relating to users, views, ports, IP's and permissions are prerequisites. Other important considerations is the fact that both teachers and learners need to apply concepts such as connecting to a server, understanding user permissions, and data relating to a schema etc.
	Both Visual Studio and Embarcadero RAD studio provides an easy interface with wizards which will allow the connection to an Access Database in simple steps without focusing on the technical details. The required connection code is automatically generated. This approach will be very beneficial in the teaching and enforcement of other DB skills with relation to problem solving.		The NetBeans IDE also provides a wizard interface for example utilizing the JDBC/ODBC bridge but it still requires some technical settings, and some inherent complex code.	** An approach utilising programming constructs beyond the scope of the CAPS such as the entity manager API (interface), which could lead to additional technical complications and overhead. [See FN1]
Query a database (single table) using simple SQL constructs	**Complexity Scale: 2 – 3** It is a relatively simple process to assign a query string to a either a data-adapter or OLE DB command.	**Complexity Scale: 2** The process is easily accomplished by assigning a SQL statement to either a TQuery or TDataset (related) component.	**Complexity Scale: 4** The Query must be assigned to the result set object. Knowledge of the methods of the result set object is required in order to allow for	**** Complexity Scale: 5** In the examples provided by the DBE training team an alternative to SQL is used to query the JPA entity objects, i.e. JPA Queries (JPQL /

	Knowledge of the Data Adaptor component or DB command object is required. The result of the query could easily be assigned to a DataGrid view component	The result of the query could very easily be assigned to a DBGrid.	functionality. In order to list the content of a query the implementation of a jTable component is required. This is an inherent complex process	Criteria) The JPA Query Language (JPQL) can be considered as an object oriented version of SQL. The main difference between SQL and JPQL is that SQL works with relational database tables, records and fields, whereas JPQL works with Java classes and objects. [See FN2] In essence the application of the JPQL requires a prerequisite knowledge of the concept of an array of objects as a <List> ans serialization [which are excluded from the CAPS]
Use programming language constructs in the execution of various simple database transactions *– Access fields and records within a dataset with code constructs and applicable methods*	**Complexity Scale: 2 – 3** In VB.NET it is required to either access the columns and rows of the DataGrid view component or to create separate table component utilising the rows property.	**Complexity Scale: 2** The records and fields within a Query or Table component are easily accessible through simple code constructs. The result dataset is easily displayed in the DBGrid component	**Complexity Scale: 2 – 3** The result set provide read only access to the results of the query selected, in a one directional manner. In order to list the content of a query the implementation of a jTable component is required. This is an inherent complex process	**** Complexity Scale: 4- 5** Here the JPA, API methods will need to be applied
Navigate the records of a dataset	**Complexity Scale: 2** Methods related to the navigation of the record pointer are available for implementation. Direct reference to the rows of the data table is also available	**Complexity Scale: 2** Methods related to the navigation of the record pointer are available	**Complexity Scale: 2** Methods related to the navigation of the record pointer in the result set	Here the JPA, API methods will need to be applied
Modify individual fields	**Complexity Scale: 3-4**	**Complexity Scale: 3**	**Complexity Scale: 3-4**	Here the JPA, API methods will need

and records within a dataset with code constructs and applicable methods, and apply all changes & *Manipulate a dataset object and records with code constructs and apply all changes* & *Use programming language constructs in the execution of various simple database transactions*	Without Using The DB command object could be implemented in conjunction with a Data Adapter component to perform record manipulation. Knowledge relating to the use of the Data Adapter as well as the DB command objects is required. All manipulation of data is to be done by the implementation of SQL constructs.	Delphi allows for the manipulation of data in a table via the invocation of simple Table methods to edit and update data in a table without SQL overhead. It also allows for the easy implementation of SQL constructs to manipulate a database table.	Java requires the use of an SQL statement object to perform a DML query to manipulate the data in a table. The method ExecuteUpdate() is required to perform the functionality. All manipulation of data is to be done by the implementation of SQL constructs.	to be applied
Incorporate dataset event handlers and methods as part of the solution	**Complexity Scale: 4 – 5** This is a relatively complex operation as it requires the implementation of Delegates where it is required to Map an Event to a specific Delegate. Delegates are in essence prewritten event handlers which must be tied to a object using code.	**Complexity Scale: 3** The TADO Table or TADO Query provides an extensive list of events which is easily accessible via the IDE. Such methods are easily incorporated into the program code.	**Complexity Scale: 5** This process is DB specific and requires the implementation of triggers or stored procedures which falls outside of the scope of the CAPS document.	Here the JPA, API methods will need to be applied
Develop a multi-form GUI incorporating simple controls *** This objective is applicable to both DB driven applications as well*	**Complexity Scale: 4** Creating Multiform or MDI applications in the .NET environment is a relatively complex process due the protective nature of the	**Complexity Scale: 3** Access to various form or MDI application forms is easily accomplished due to the composition of the VCL architecture.	**Complexity Scale: 5** Creating Multiform or MDI applications in the NetBeans environment is a relatively complex process. Multiple JFrame objects need to be instantiated with the	Here the JPA, API methods will need to be applied

as non DB driven applications and scenarios	environment with relation to form objects. Extensive knowledge on reference passing is required to share form class object properties and events.		manual inclusion of accessor methods to share various objects (components) attributes in the same multiform application. Properties are also not directly assessable – See note on page 5.	

FN1 – JPA training

JPA training is provided to candidates which are already reasonably knowledgeable in standard Java programming which include: Java Developers, Java EE Developers (learner will only have 6 month's basic Java programming experience)

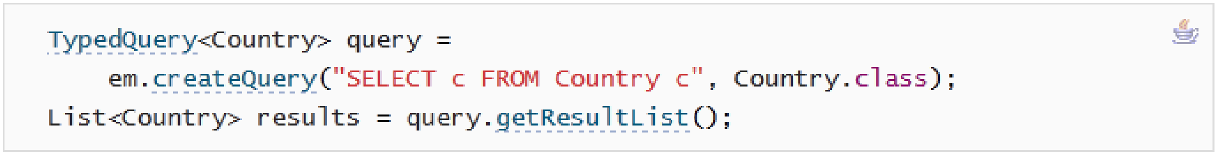
Required Prerequisites

- Display experience with the Java programming language
- Integrate existing Java code (for example, reuse existing classes created by other team members)
- Java Programming Language, Java SE 6

(http://education.oracle.com/pls/web_prod-plq-dad/db_pages.getpage?page_id=609&p_org_id=30&lang=US&get_params=dc:D65187GC10,p_preview:N)

FN2 – Code complexity

The ability to retrieve managed entity objects is a major advantage of JPQL. For example, the following query returns Country objects that become managed by the *EntityManager em:*



```
TypedQuery<Country> query =
    em.createQuery("SELECT c FROM Country c", Country.class);
List<Country> results = query.getResultList();
```

http://www.objectdb.com/java/jpa/query/jpql/select

FN - General

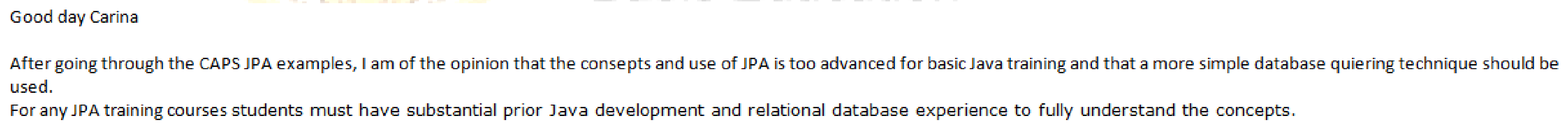
The content and prerequisite knowledge required surpasses the intended application of the CAPS. To fully understand the use of the JPA, which is a complicated architecture based on an abstraction layer requires detailed understanding, to grasp some basic concepts.

The approach for Java presented as part of the CAPS training, is cognitively challenging, and could have some negative impact at later stages with regard to the API versioning.

http://www.objectdb.com/java/jpa/persistence/managed

http://pic.dhe.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=%2Fcom.ibm.websphere.ejbfep.multiplatform.doc%2Finfo%2Fae%2Fae%2Fwelc_newinreleaseejbfp.html

Comments from senior Java Developers in industry:



Good day Carina

After going through the CAPS JPA examples, I am of the opinion that the consepts and use of JPA is too advanced for basic Java training and that a more simple database quiering technique should be used.
For any JPA training courses students must have substantial prior Java development and relational database experience to fully understand the concepts.

Hi Carina,

After reviewing some of the content as presented as part of the CAPS IT Grade 11 and 12 training material due to a request by a colleague we would like to point out the following concerns relating to the Java aspects of the curriculum:

- Java DB document
  - Derby has an incomplete support for full SQL vocabulary
  - Derby supports basic index's only (B-/B+ trees only) – effectively no specialized indexing
  - Never used in enterprise class or professionally delivered software solutions
- JPA Document
  - Only useful for very limited / narrow scope scenario as object based persistence enforces a single hierarchical view of the underlying data
  - This is not very useful in today's data driven economy as many different views of the same data is required for more and more integrated applications – constraining the underlying entity model in the RDBMS to an application specific object signature limits the use of the data and is very rarely used in industry
  - Takes away from the importance to know and understand SQL as this is once again becoming more and more important, especially with the drive towards in memory computing where a lot of logical processing is moved to the database and accessed with extended SQL syntax – without a foundation of basics SQL one will be at a distinct disadvantage when entering this arena
- Java vs Delphi
  - Delphi is syntax and environment stable – Java is not – Learning runtime environment dependencies and configuration (JRE's, Class paths, Class version dependencies and conflicts, Java container's and their individual features, characteristics module loading techniques etc.) goes far beyond what any secondary school pupil should be exposed – Delphi is a static and stable language and runtime and these concerns are for the most part negated
  - Delphi reads easy and is syntactically more easily linked to human readable logical content of a program than Java which requires a lot of in line commenting to convey human readable logical content and for that reason alone is probably not suited to a secondary school classroom environment where the conveying and grasping of high level abstract programing concepts and notions should take precedence.
  - With Java there is an additional layer of complexity between the concepts and the implementation (which solidifies a pupils grasp of the concept). The complexity of translating abstract concepts into practical implementation is much reduced with Delphi syntax, inherently stable runtime environment and the simplicity of dependencies, reduced complexity and speed of compilation. Even though Delphi is much less suited to complex, large scale highly integrated enterprise solutions it is much better suited to classroom code based representation of generic abstract programing concepts.

ANNEXURE C

1. The choice of a first high-level programming language in schools and other educational institutions is a contentious issue that has been debated for decades and often lead to emotional, almost childish debates. The article written by Siegfried, Chays and Herbert (2008), *Will there ever be consensus on CS1?* starts with the following abstract:

   The choice of programming language, the approach by which students are taught and the software tools made available to students have been controversial issues in many ways. While there once was a consensus of some sort within the computer science education community, it is much more difficult to find common ground among those of us who teach introductory programming courses. The literature is explored and answers sought to the question of which language is optimum in teaching novice programmers, as well as the approach that ought to be used. Finally, the question of whether a consensus can be reached is addressed.

   The paper further argues that most languages that are presented for use in a CS1 course have some aspects that make them undesirable to some faction within the computer science education community. The paper concludes:

   Arguably, the discipline may need a new teaching language that will offer the benefits that the computer science education community found in Pascal over thirty-five years ago. But at the present, it seems that there will be great difficulty finding that consensus.

2. The article, *Choosing a First Programming Language*, by Randy Kaplan (2010) states the following:

   When choosing a programming language to teach students as their first programming language, which one should be chosen? There are approximately 2000 to 3000 known programming languages documented on the World Wide Web. Which one would be the best to

   (1) teach students the proper concepts of programming, and

   (2) maintain student's interest in programming as an aspect of computer science?

   It concludes:

   It seems reasonable to suggest that it is time for a novice programming language to be created that is designed around not only technical concepts but also educational and psychological concepts.

   The DBE therefore focused on the two languages currently used (Delphi and Java) as well as the alternative one suggested (VB.Net).

3. The article, *Will there ever be consensus on CS1?* states the following:

   Java is not an ideal language for beginners. McIver points out that Java's modular structure and requirement that every data item and method be part of a class mandate a certain minimum size for every program, no matter how simple it may be. This also applies to the definition of constants, which can require as many as four reserved words. While a subset of Java can minimize the problems that novice programmers must face, it is very difficult to create a subset that addresses all these concerns. The popularity of Java is partially due to the fact that it is used for many real-world applications, particularly web-related applications. Yet there are many features that make it difficult for novice programmers.

McIver examined several languages, including Java, which failed to meet the optimal case for cognitive dimensions. Pascal remains the closest to optimum of the four languages shown.

Similar arguments are raised in other literature.

4. Furthermore, it should be noted that the Assessment and Qualifications Alliance's (AQA) in the UK announced in 2010 that A-level computer science students will no longer be taught C, C# or PHP from next year (2011). The following is an extract from the statement in *The Register*:

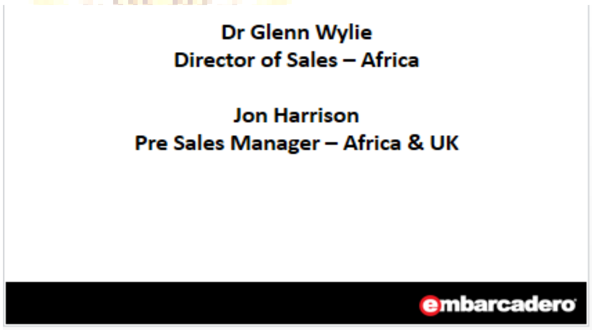
   The board "highly recommended" switching to Pascal/Delphi because it is stable and was designed to teach programming and problem solving. Teachers planning to use Java are warned that many universities are considering dropping it from their first year computer science programmes, "as has happened in the US".*

   The document detailing the withdrawal, states:

   Pascal/Delphi is highly recommended because it was designed specifically to teach programming and problem solving - see http://uva.onlinejudge.org/ - and it is stable. Its event-driven forms-based object Pascal manifestation, Delphi, has excellent support for a range of applications from networking through graphics to databases. Delphi is still rated as the world's best RAD system and is used extensively throughout the world for commercial application development.

5. It should be noted that there is not such a richness of literature available with regard to Delphi and education per se. Most of the articles focus on Pascal. Therefore, these were used as a basis to evaluate Delphi (Object Pascal), also considering the added features and functionality. The focus of the evaluation is educationally sound criteria of which readability/simple syntax and ease of use, especially considering the requirements of the IT CAPS were highly valued.

About Delphi and Embarcadero

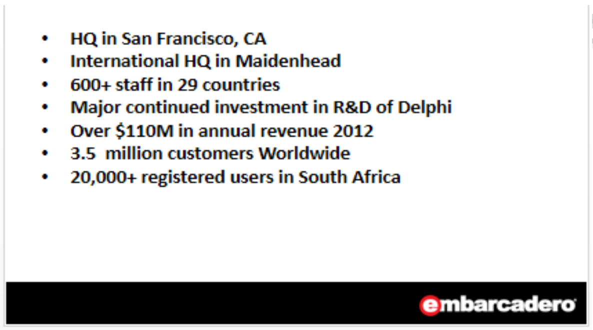


Dr Glenn Wylie
Director of Sales – Africa

Jon Harrison
Pre Sales Manager – Africa & UK

embarcadero

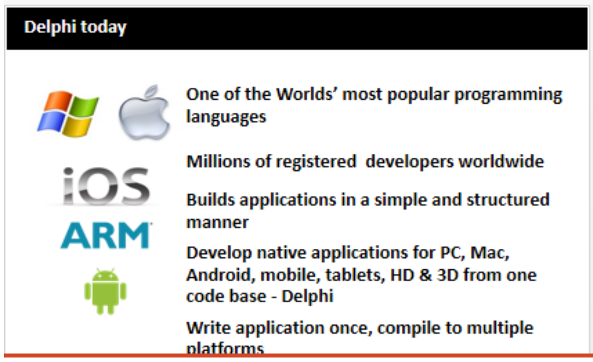
1



- HQ in San Francisco, CA
- International HQ in Maidenhead
- 600+ staff in 29 countries
- Major continued investment in R&D of Delphi
- Over $110M in annual revenue 2012
- 3.5 million customers Worldwide
- 20,000+ registered users in South Africa

embarcadero

2



Delphi today

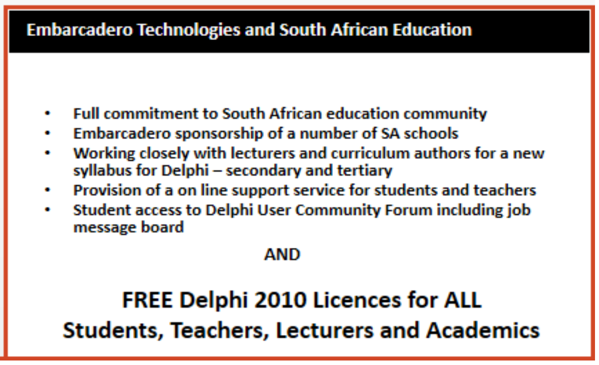
One of the Worlds' most popular programming languages

Millions of registered developers worldwide

Builds applications in a simple and structured manner

Develop native applications for PC, Mac, Android, mobile, tablets, HD & 3D from one code base - Delphi

Write application once, compile to multiple platforms



Embarcadero Technologies and South African Education

- Full commitment to South African education community
- Embarcadero sponsorship of a number of SA schools
- Working closely with lecturers and curriculum authors for a new syllabus for Delphi – secondary and tertiary
- Provision of a on line support service for students and teachers
- Student access to Delphi User Community Forum including job message board

AND

FREE Delphi 2010 Licences for ALL
Students, Teachers, Lecturers and Academics

Summary of aspects that could be considered:

Aspect for consideration	**Delphi**	**Java**
Examination	More than one programming language, different versions, databases, query language - problematic	
Migration	Learner migration between Delphi and Java provinces is problematic	
Resources **(Textbook)**	No Grade 11 IT textbooks listed on national catalogue Textbook (not evaluated by DBE) available for Delphi	No textbook available for Java – problematic as CAPS follows new approach
Support	Resources needed in two languages.	
Training	**Java** teachers will need **Delphi** training	
IDE/GUI builder	Integrated with Delphi (RAD studio) – no additional software required – seamless integration	**Java** requires Netbeans – not integrated but a bolt-on – could be problematic especially if correct versions/combinations are not used
Database	**MS Access** for Delphi – not server-based (bolt-on). No problems experienced – have been using Access and Delphi since 2000 (Integrated server-based database available that also works seamlessly)	**Java** uses built-in JPA (server-based) – designed for industry use and could be very complicated for learners as it requires additional overhead concepts not required or specified in curriculum such as knowledge of Client-Server computing, SQL DML scripting etc.
Educational soundness	**Delphi** is easier to read, closer to natural language, less syntactical overhead More suitable as a first language (comes from Pascal which was designed for teaching programming) UK (AQA) recommended **Delphi** for Computer Science courses	**Java** syntax more difficult to read, more syntactical overhead Many are phasing it out in introductory courses Questionable w.r.t. suitability for learners Even some Java developers in industry raised their eyebrows in learning that Java is used as language in schools Literature suggests that Pascal (Delphi) is more suited than Java and states that *despite the popularity of languages such as Java, C and C++, there has been much debate about the suitability of these languages for education, especially when introducing programming to novices*

Aspect for consideration	Delphi	Java
Industry relevance	**Delphi** not as widely used as Java, though rapidly growing with exciting new features	**Java** is very widely used in industry
Technical issues	Backwards compatibility in **Delphi** is normally addressed very easily	As **Java** is open source, stability could be questioned. **Java** users will always have to ensure that they use the same JDK version (and the latest version) else compatibility issues might arise (as we have seen in the 2010 matric exams in terms of the data files as well as memo discussion with regard to different classes used for input, etc.) – not always fully backwards compatible The same applies to Netbeans
IT CAPS requirements	No problem with **Delphi –** best suited for CAPS (recommended by DBE)	Some aspects of the curriculum (databases) harder to implement in **Java** and more complicated in Java which requires the use of concepts outside the curriculum, e.g. use of record lists as containers when dealing with the database. The inclusion of the Java Persistence Query Language which allows for queries over objects.
Cost	**Delphi** is not free Embarcadero announced free Delphi 10 licenses for schools for limited period Embarcadero offers educational licences with a 90% discount on the normal licence fee.	**Java** as well as Netbeans are open source and therefore free Support for open source software might be a problem Many different versions are problematic as compatibility problems between different versions exist Features used today to implement the curriculum may not be available with new versions

ANNEXURE C –

Why choose Delphi/Object Pascal for GCE Computing - Assessment and Qualifications Alliance (AQA) – UK

(Copied from original PDF document)

Teacher Resource Bank

GCE Computing

Other Guidance:

 Why choose Pascal for GCE Computing?

Why choose Pascal? Industrial-strength, fashionable languages versus languages for teaching the principles of computing and programming

AQA's advice that teachers choose to teach their candidates Pascal for the GCE in Computing, rather than any other programming language is based on sound pedagogical reasons supported by current research.

Quoting from one among several papers:

"*Firstly, a computing curriculum should not become a vocational training ground for current industrial-strength programming languages and programming tools. Any introductory course in Computing should not be arranged around the syntax of a currently fashionable programming language. It is more important to concentrate on principles. At the same time, however, a curriculum must also teach how these principles apply to the real world, but this relates to teaching program design principles not the use of language constructs. Teach good habits early otherwise bad habits become ingrained and require costly fixes. To avoid any confusion, the course should not use a complex language that distracts from design principles and should not pose problems from complex application domains. The first language should facilitate the teaching of design principles.* " The Structure and Interpretation of the Computer Science Curriculum, Matthias Felleisen et al, Journal of Functional Programming(2004), 14: 365-378 Cambridge University Press.

Java, C, C++, C# fall into the category of complex programming languages. The design principles that should be covered are data abstraction, functional/procedural abstraction, data-directed programming.

What constitutes an appropriate programming language for introducing principles of programming?

Firstly, the language must be one that supports the exploration of algorithms and principles of computation.

Secondly, the language must support the teaching of program design principles.

Thirdly, it must support progression to teaching OOP principles in a user-friendly way and be capable of event-driven programming in a visual environment.

Fourthly, it must have good support for creating database applications and networking.

Fifthly, it must be extensible so that other types of application programming such as 2-d and 3-d gaming can be supported.

The language must be one that supports the exploration of algorithms and principles of computation.

Pascal is a very easy programming language to understand and algorithms continue to this day to be expressed in Pascal-like pseudo-code. There is now plenty of research literature that highlights the difficulties that university Computer Science/Computing departments have experienced introducing programming to novices. Universities switched from teaching introductory programming courses using Pascal to using C which they then abandoned as being too difficult, very quickly. C was replaced by Java with similar adverse results. Many Computer Science/Computing departments have now switched to **mini-languages based on Pascal-like languages** or functional model-view programming languages based on Scheme for introducing the principles of programming. Some of these have adopted a visualization approach using microworlds, e.g. C-Sheep uses a subset of C but it resembles Pascal very strongly- The National Centre for Computer Animation, Bournemouth University (URL: http://ncca.bournemouth.ac.uk/eanderson/C-Sheep/), “Karel the Robot” is also very popular. Its syntax is based on Pascal.

The language must support the teaching of program design principles.

A paradigm shift has also occurred with a return to using a procedural language approach for introducing programming to novices rather than an object-oriented language. The teaching of industrial-strength languages such as Java is now being delayed until later in undergraduate courses. Interestingly, Oxford Brookes teaches Delphi to their undergraduate first year students, beginning with Pascal console mode within Delphi and later switching to the event-driven, object-oriented programming environment within Delphi. Oxford Brookes entry requirements for their Computing courses are lower than the Russell group universities.

It must support progression to teaching OOP principles in a user-friendly way and be capable of event-driven programming in a visual environment.

Delphi is a robust and commercially successfully language used extensively in North America and Europe. In fact, the support in Europe in education is extensive. Hence the Lazarus project, an open source initiative that supports Pascal and Delphi. Lazarus Pascal and Delphi is recommended for use in schools in mainland Europe. Whilst on the International front for our brightest computing students to compete in the International Olympiad of Informatics (IOI) they must know either Pascal or C. The lack of educational opportunities pre-16 to learn the principles of programming means that potential IOI students from UK have to start from scratch at sixteen. Opportunities to compete nationally and then internationally therefore favour the teaching of Pascal.

It must have good support for creating database applications and networking and it must be extensible so that other types of application programming such as 2-D and 3-D gaming can be supported.

Delphi has been voted the world's best Rapid Application Development (RAD) environment because of its excellent integration with a range of databases and also excellent networking solution support. It is also easily extensible with components that support a range of other applications including 2-d and 3-d gaming. Interestingly, a very popular C/C++ development environment, Visual Dev-C++, is written in entirely in Delphi. Candidates using Delphi may aspire to producing very complex applications eventually. We already see a very high standard of project work amongst the candidature from centres that teach Pascal/Delphi.

Visual Basic versus Pascal/Delphi

Although Visual Basic is very popular, it is considered a poor language for teaching principles of Computing. Very few university Computer Science/Computing departments teach Visual Basic. This is underpinned by research that has compared using Visual Basic with other languages. This research clearly makes a solid case for using Pascal or Pascal-like languages rather than Visual Basic to introduce programming because it is based on an event-driven programming environment.

Microsoft made the decision sometime ago to withdraw support for Visual Basic 6 as it concentrates on Visual Basic for .Net which is not compatible with VB6. Its replacement C# borrows very heavily from Pascal and indeed its designers were the same team that designed Delphi. However, C# is considered an inappropriate language for introducing the concepts of programming to novices. We acknowledge that many centres use Visual Basic, currently. We intend to continue to support these centres if they wish to continue with using Visual Basic. However, we also wish to encourage centres to consider adopting a language that is more suited to teaching computation. Currently, we feel that this language is Pascal/Delphi although in the future it could very well be another language such as Python. The feedback that we have got from universities supports this view.

Borland have made versions of Pascal and Delphi available at affordable prices for use in centres and free versions for personal use are available. The Open Source Lazarus project which supports the teaching of Pascal and Delphi in European schools is another source.

Bibliography

P. Brusilovsky, E. Calabrese, J. Hvorecky, A. Kouchnirenko, and P. Miller.

Minilanguages:

A way to learn programming principles. Education and Information Technologies, 2(1):65{83, 1997.

L. Carter. Why students with an apparent aptitude for computer science don't choose to

major in computer science. ACM SIGCSE Bulletin, 38(1):27{31, 2006.

P. Brusilovsky, E. Calabrese, J. Hvorecky, A. Kouchnirenko, and P. Miller.

Minilanguages:

A way to learn programming principles. *Education and Information Technologies*, 2(1):65–83, 1997.

D. Buck and D. J. Stucki. Jkarelrobot: a case study in supporting levels of cognitive development in the computer science curriculum. In SIGCSE '01: Proceedings of the thirty-second SIGCSE technical symposium on Computer Science Education, pages 16–20, 2001.

L. Carter. Why students with an apparent aptitude for computer science don't choose to major in computer science. ACM SIGCSE Bulletin, 38(1):27–31, 2006.

S. Cooper, W. Dann, and R. Pausch. Alice: A 3-d tool for introductory programming concepts. *Journal of Computing Sciences in Colleges*, 15(5):107–116, 2000.

W. Dann, S. Cooper, and R. Pausch. Making the connection: Programming with animated small world. In Proceedings of the 5th annual SIGCSE/SIGCUE ITiCSE conference on Innovation and technology in computer science education, pages 41–44, 2000.

T. Jenkins. The motivation of students of programming. In ITiCSE '01: Proceedings of the 6th annual conference on Innovation and technology in computer science education, pages 53–56, 2001.

Do Robots Dream of Virtual Sheep: Rediscovering the "Karle the Robot" Paradigm for the "Plug Play Generation", Eke Falk Anderson, eanderson@bournemouth.ac.uk, Leigh McLaughlin, lmcloughlin@bournemouth.ac.uk, The National Centre for Computer Animation, Bournemouth University, Talbot Campus, Fern Barrow, Poole, Dorset BH12 5BB, UK