

Runtime Environments – the Business Case

Roy Ben Hayun

February 2007

1 INTRODUCTION	2
2 RUNTIME ENVIRONMENTS ENABLE STRATEGIC GOALS	2
2.1 CREATE AND INCREASE REVENUE STREAMS	2
2.2 SPEEDING UP TIME TO MARKET (TTM)	3
2.3 ENHANCING THE SYMBIAN OS RICHNESS	4
3 TECHNICAL DECISION MAKING PROCESSES	5
3.1 TAKING THE TECHNICAL DECISION	5
3.2 HOW THE DECISION IS MADE	5
3.3 WEARING DIFFERENT HATS	5
3.4 ROBUST TECHNICAL AND BUSINESS PARADIGMS MIX	6
3.5 TECHNICAL ORIENTATION WIN CASES	6
4 SYMBIAN'S RUNTIME ENVIRONMENT STRATEGY	6
5 NEXT TIME	7
6 GLOSSARY	7

1 Introduction

This is the first instalment of Mobile Runtime Environments, which gives developers, technical leaders and technical decision-makers a glimpse of the various development platforms available for Symbian OS.

A Runtime Environment can be defined as a non-native execution platform that is hosted by and receives services from the underlying native Operating System. Well known examples for Mobile Runtime Environments are Java, Flash, Ruby, Python and more.

The focus of this month's article is the business justification for supporting and enabling a variety of Runtime Environments. We will first explore the link between strategic technical decisions (and Runtime Environments specifically) and strategic company goals. Secondly, we will look at company processes involved in technical decision-making and Runtime Environments. Finally we will see what Symbian's Runtime Environments strategy is.

2 Runtime Environments help meet strategic goals

For a Runtime Environment to be chosen as the development platform for a software product, it needs to satisfy requirements arising from solid business-oriented thinking. The technical aspects may well play a part in the choice but the decision makers, who will determine which Runtime Environment to go with, are unlikely to be swayed by theoretical beauty, richness of language features or other computer language philosophies. They have a business to run, products to develop and customers to satisfy. They have strategic goals.

The basic key criterion in every product, how well it will sell, applies to computing languages and Runtime Environments as well. Otherwise, the language or Runtime might be fascinating theoretically but if it is not applicable in the day to day software engineering world, it is unlikely to be hosted or have a developer community.

Computing languages and Runtime Environments evolved out of a need to achieve the strategic goals of powerful stakeholders in the industry, i.e. network operators, device manufacturers, ISVs, content providers, etc. Apart from the large number of Runtime Environments that exist, some of them have established themselves as leading platforms, and from examining the Runtimes that established such market leadership, it is clear that they indeed enable strategic goals.

The following three sections examine what those strategic goals are and how Runtime Environments assist in achieving those goals.

2.1 Create and increase revenue streams

Stakeholders are constantly seeking to create revenues in areas which have been identified as proven revenue streams, in addition to increasing their revenues in existing areas.

Any Runtime Environment which helps these companies to achieve their goals would be considered a candidate development platform.

As an example, let's look at gaming. In this scenario, an organisation could make a strategic decision to go into the gaming, which it has identified as a profitable revenue stream. Alternatively, a company which is already in gaming could decide to increase its existing revenue stream by expanding into mobile platforms or other specialised gaming arenas such as mobile 3D games.

Having identified its strategy, let's discuss how such a company would select its Runtime Environment by looking at the technical features provided by Java. It is clear that, by adding the gaming API to the MIDP 2.0 specification and bringing gaming support and capabilities to the

JavaME platform, a new revenue stream was made available for many stakeholders in the industry. In the same way, 3D support in JavaME provides another option for the companies who want to expand their gaming revenues by joining the 3D gaming niche.

We should note that JavaME, or any other Runtime Environment, is not in itself a revenue stream and does not create one. Java technology has been harnessed to deliver services which have been identified as a future or existing revenue stream. In defining the MIDP2.0 variant, the goals of the industry group behind it was to take Java technology and harness it to deliver compelling services which are strategically important for many service providers.

It is unlikely that anyone is willing to pay for an application solely because it was written in Java. But a lot of people are prepared to pay to buy a game and now a stakeholder can use Java technology to deliver gaming products on a wide range of mobile devices and to deliver services much quicker than using native development technologies.

In fact, Java has established itself as the number one wireless gaming platform in volume, quality and deployments, and commercial mobile games (source: Wireless Gaming Review).

Another reason for JavaME's success is that it enables companies to gain a wide penetration into the mobile devices applications market. As a popular hosted execution environment, it provides an easier and wider access into this market than native development technologies. For example, it is not necessary to develop a native application for each model or platform (at least not in the purist sense) and this provides a larger potential market which increases the potential revenue stream.

2.2 Speeding up Time to Market (TTM)

Every company wants to decrease the time taken for its products to reach the market so that its products are announced before those from competitors, and to reduce the development costs of the products.

Since all companies follow these basic principles, the whole IT industry is operating in Bullet Time – ship your product tomorrow or another company will be there at the beginning of next week.

Using Runtime Environments helps to achieve faster TTM in two aspects of software projects – people and technology.

Scott McConnell wrote in his book *Rapid Development*: “... *your software project operates along four important dimensions: people, process, product and technology. People perform quickly, or they perform slowly.Technology assists the development effort, or it thwarts developers' best attempts*”

Starting with the technical aspect, high level languages have been designed with ease of development in mind and include language features and attributes that assist in creating rapid solutions. High level Runtime Environment libraries serve the same purpose by providing developers with built in services to minimize development effort. Thus development on top of high-level Runtime Environments achieves the required results with less effort and costs.

Additionally, Runtime Environments have an advantage over native proprietary platform development in their ability to share or reuse code, as discussed below

An example for sharing code in client-server projects is a JavaME-JavaEE project with proprietary code that can be shared between the server and client and save separate development (as compared to development in Symbian C++ which cannot be deployed on the server side).

In the same way, reusing code saves development effort, so when a new project starts you can utilise existing components instead of recoding from scratch. Given the existence of such content or code, in a smaller porting effort (i.e. not on the scale of re-implementing an existing program in C++) it would run almost unchanged on Symbian OS using the equivalent Runtime Environment.

For example, porting Flash content which was created for desktops, to run on Flash-Lite does not generate new content but re-uses the existing content.

That is how Runtime Environments assist the development effort. Moving to the people aspect, an advantage of a Runtime Environments is its developer community and the large number of skilled developers already on that platform.

Having the Runtime Environment hosted on Symbian OS turns the whole developer community for that environment into potential productive contributors to the ecosystem. Those engineers can become productive immediately, producing content for the Runtime Environment hosted on Symbian OS. Later on, those skilled developers could, if applicable, expand into other domains of Symbian OS development – on other Runtimes and in Symbian C++, through a more gradual transition. For example, instead of jumping from Python on desktops straight into Symbian C++, they could first move to the Symbian OS, on a development platform which they are familiar with, and later on move to native development if that is applicable or required.

Additionally, it is typical that high level languages would have a shorter learning curve and it takes less time for engineers to become productive on that Runtime Environment, instead of requiring full blown mentoring in native Symbian OS idioms.

In both cases, you get people to perform quickly.

2.3 Enhancing the Symbian OS richness

A rich platform is crucial in enabling ecosystem stakeholders to develop their solutions. Runtime Environments play a major strategic role in enriching the platform with the content, opportunities, IP and developer community that they bring to the Symbian ecosystem.

Hosting a Runtime Environment on Symbian OS will attract to its developer community, the talent and knowledge of software engineers from other ecosystems. For example, with the port of Ruby to Symbian OS, Ruby developers have an easy migration path into the Symbian ecosystem through a Runtime they are familiar with from previous experience on other hosting platforms. There is no need for them to learn native idioms and go through a steep learning curve, they can become productive immediately and harness their talent, knowledge and experience, thus mitigating the risk in which content and IP for Symbian OS comes only from inside the OS world and none from outside.

The choice of development options is an important factor for stakeholders and hosting a variety of Runtimes on Symbian OS will give more options to those stakeholders who want to be able to be able to pick. Having this variety will improve their chances of finding a fit-for-purpose development solution and will enlarge the pool of candidate developers who can help them in the product development.

Even if the product is not implemented in any hosted Runtime Environment, there are plenty of development tasks which could be implemented in a non-native Runtime Environment and thereby reduce the development effort. For example, even in the case of a native application, it is possible to do quick prototyping with Python on the same device or to create a logging utility for debugging purposes implemented in Ruby or Perl.

Another reason for Runtime Environment is... fun!

Developers want to have fun and we must provide a bigger technical playground in which developers can explore different options, enhance their technical skills and have some fun too.

3 Technical decision making processes

Having looked at the strategic goals, it is natural to continue the discussion by investigating how technical decisions are taken in different organizations. By discussing this, we will gain more knowledge in the processes through which a Runtime Environment might be used as a strategic development platform.

The technical perspective and business perspective have to live together in the real world, and their co-existence is interesting to observe.

Looking at how companies make technical decisions and how the business perspective is involved, or the types of companies who make decisions in a more technical or more business point of view, would help in the understanding of how multiple paradigms co-exist.

Note that, as the scope of the series is Runtime Environments, our discussion of concepts, units and processes will be confined to that context.

3.1 Taking the technical decision

In the majority of companies, technical decisions are meant to assist the business strategy of the company. The technical organizational unit will usually act as adviser and consultant to the decision making organization unit which will have a business orientation. It could even be that the two units are merged into a single unit.

The influence that the technical side of things has on the business decision really depends on the culture of the organization as well as the competitive environment. For example, Intel is known for having a very strong technical leadership and so the decision to leave memory DRAMs was driven from below by engineers not by top management. On the other hand, Applied Materials has a powerful regional leadership and technical product developers need to convince head of regions who are marketing people to use their product.

From that reason, the Runtime Environments series has started with a less technical perspective. But be assured we will be back in the technical arena next time.

3.2 How the decision is made

In a hard-and-fast simplistic categorization of big, medium and small organizations, the size would be a significant factor in the decision process, including decision processes that are not explicitly seen as a defined written down process.

In the smallest start-ups, the decision maker could be a single developer or an experienced entrepreneur (with the Venture Capitalist having their own opinion). In medium and large companies the decision could then be taken by a larger group of opinion holders. In the large companies, the official decision is often taken by executives at a senior level who ideally have sufficient knowledge on each of the domains.

Choosing a Runtime Environment as a development platform works in the same way. It might be a single developer in a small company, a decision by a group of experts and also a legacy platform from earlier days of the company.

3.3 Wearing different hats

A Runtime Environment needs to satisfy different requirements and success criteria and not just the technical orientation.

If we look at just a few different organizational roles, seeing what they emphasize, or define, as success is a good way to provide a concrete example of the application of different orientations.

What makes the project manager happy is to go home every day confident that the metrics are showing a positive trend toward the predicted end of project. What makes the business entrepreneur happy is to return the investment in the short term and to grow the business in the long term. What makes the technical manager happy is to go home every day confident that his project is showing a positive trend in the metrics of functional and non functional requirements.

Naturally there are more roles and hats to wear, but these examples are sufficient to show the different definition of professional success and how Runtime Environments need to satisfy various requirements and not just technical ones.

3.4 Robust technical and business paradigms mix

What is important for a business is maintaining a healthy balance between the technical, academic and business paradigms, which should be evaluated together with a view of the goals and constraints.

A decision made with a purist technical orientation or a purist business orientation is likely to be a single-minded result and as such could be vulnerable to risks which arise from the reality of the other, lesser thought about, domain.

However, in a non-predictable environment, a single minded orientation might win with a full sweep against all those risks and provide the entire holy grail list of goals.

Like any other solution, the Runtime Environment needs to maintain a healthy balance between different factors such as being fit for purpose technically, being suitable and applicable for creating consumer products and many other factors.

3.5 Technical orientation win cases

Certain technologies sound great. They talk the talk, walk the walk and eventually sell well. They are cool in the eyes of developers and everybody can see the cash on the table. An example of these are those technical trends with the likeability factor that wins the hearts and minds of people holding either a technical or business perspective.

The business entrepreneur has heard about them from his colleagues and therefore they are perceived as safe and reliable for investment. The technical manager understands how they became so successful from the insider knowledge he has in this domain.

In many cases, Java is such a technology. It sounds great, has the likeability factor and can be seen as a model for technology and business creation (Having said that, it does not mean Java is the right solution for *your* product).

4 Symbian's Runtime Environment strategy

So we see that Runtime Environments can serve a number of important purposes. What does Symbian do about them?

Essentially, Symbian's strategy is to enable licensees and third parties to deliver competitive Runtime Environments running on Symbian OS by providing native APIs as needed and to provide generic changes to existing components as required.

Symbian does not push any one Runtime Environment ahead of others. Symbian's role as the centre of the ecosystem is to provide an open platform that enables development platforms which

give opportunities to developers. Sometimes Symbian has chosen to actually implement a Runtime Environment (i.e. OPL, PJava and J2ME) or to create a limited size project (i.e. Ruby for Symbian OS), and in other cases to just enable it, thereby ensuring that a Symbian partner who wants to implement the Runtime Environment can do so.

The width and depth of available technologies and rich APIs, which make Symbian an industry leader, is what gives the hosted Runtime Environments the potential to be very powerful technically and assist in achieving stakeholders' strategic goals.

5 Next time

In the next instalment we will say goodbye to strategic goals and dive deep into the fascinating technical world of computing languages, computation libraries, system libraries, hosted execution environments and more.

The topics will involve concrete examples from all major Runtime Environments like Java, Ruby, Python, Flash and friends.

And it's going to be techy. Very techy.

6 Glossary

The following technical terms and abbreviations are used within this document.

Term	Definition	Reference
JavaME	Java Mobile Edition	http://java.sun.com/javame/
MIDP2.0	Mobile Information Device Profile	http://java.sun.com/products/midp/
JavaEE	Java Enterprise Edition	http://java.sun.com/javaee/
Ruby		http://developer.symbian.com/main/tools/opensrc/languages
PJava	Personal Java	http://java.sun.com/products/personaljava/
Python		http://developer.symbian.com/main/tools/opensrc/languages
Flash		http://www.adobe.com/products
Flash Lite		http://www.adobe.com/products/flashlite/
OPL		http://developer.symbian.com/main/tools/opensrc/languages
Perl		http://developer.symbian.com/main/tools/opensrc/languages

[Back to Developer Library](#)

Want to be kept informed of new articles being made available on the Symbian Developer Network?

[Subscribe to the Symbian Community Newsletter.](#)

The Symbian Community Newsletter brings you, every month, the latest news and resources for Symbian OS.