Advantages of a Software Development Framework

In this article, we define a software development framework as comprising a particular set of rules, design patterns, interfaces, libraries of modules and tools that facilitate software development, maintenance and reuse. Specifically, a development framework enables software developers to develop quality software efficiently. Moreover, software developers can easily maintain software developed by others because it is easy to understand the role, function or position of a specific module in the developed software.

The key benefits of framework-based software development include:

- **Enhanced development productivity**, since developers can focus on the business logic as the framework provides common modules and standard templates that minimize duplication of work
- **Increased component reusability**
- **Increased interoperability** by using standard interfaces
- **Standardization** of development source code.

---

1 Omar A. El Sawy is the accepting senior editor for this article. An earlier version won Second Prize in the 2012 SIM Best Paper Award Competition and was presented at SIMposium 2012.

2 The authors wish to thank Young-il Kwon (Executive Director, National Information Society Agency) and Eunju Kim (Director, National Information Society Agency) for their cooperation and support.
Lessons for Software Development Ecosystems

The benefits of a framework-based development method compared with traditional development methods are depicted in Figure 1.

A proprietary software framework is developed by the research and development (R&D) department of a software company for use within the company, and acts as an internal communication tool among developers and managers. Externally, a proprietary framework is seen by competitors and clients as a “black box” that makes it hard for them to maintain the developed software. An example of a proprietary black box software framework is Salesforce.com’s Force.com. Although Force.com encourages software developers to build complementary applications, it is not based on open source; rather Force.com is designed specifically to build applications to run on Salesforce’s CRM platform. Moreover, Salesforce runs AppExchange, a marketplace for businesses and developers to source or market applications based on Salesforce’s proprietary framework.

Software firms with a proprietary framework often collaborate and compete with different parties, thereby forming an ecosystem for mutual benefits. The development framework, or a common platform, enhances the ecosystem by facilitating the sharing of information and knowledge, enabling those within the ecosystem to have a competitive advantage over external parties not part of the ecosystem. For example, a platform ecosystem that includes independent software vendors enables them to extend their market reach and increase sales.

In contrast to a proprietary framework, an open source development framework provides access to source code (open software), offers software at no cost (free software) and allows modifications to be created (free licensable architecture). Currently, it is common for a software vendor’s ecosystem to be based on a proprietary framework, which is open to some extent to enable external developers to contribute. However, the pool of potential developers tends to be smaller than with an OSS development framework and, as a consequence, the pace of enhancing software modules is likely to be slower.

Software companies and individual developers are increasingly adopting open source software for two main reasons. First, the OSS community has grown significantly over the years, and the OSSF is not only a source of innovation but also an important source of software engineering research. Second, the OSS movement promotes an open culture and fosters collaboration, which is crucial for the development of high-quality software.

Figure 1: Benefits of a Framework-based Development Method

(Source: National Information Society Agency, South Korea)

years, with organizations such as SourceForge and Black Duck providing companies with information about open source software. Second, the release of free software libraries such as Spring and Eclipse makes it easier for companies to assess and use open source software. Further, software companies are increasingly adopting software reuse practices to improve software quality and development efficiency. In general, large software companies tend to have software product lines and development frameworks that enable formalized reuse of software components. In contrast, small and medium software businesses tend to reuse software on an ad hoc basis as they often lack development frameworks that facilitate reuse.

In this article, we examine the experiences of South Korea's government in developing an open source development framework called eGovFrame. The eGovFrame ecosystem can provide important lessons for software vendors' ecosystems. A conceptual depiction of the eGovFrame OSS framework versus the proprietary Force.com software framework is shown in Figure 2.

**OSS Development Models**

OSS development methods are generally classified into two governance models—Cathedral and Bazaar. With the Cathedral model, the source code is available with each software release, but code developed between releases is restricted to an exclusive group. An example is GNU Emacs. With the Bazaar model, code is developed over the Internet and is publicly available. An example is the Linux operating system. OSS development has evolved to become less Bazaar-like. Termed OSS 2.0, this type of development emphasizes a strategy for market creation and applications in business (vertical) domains. Even though many open systems projects exist outside of formal organization structures, and OSS communities such as GitHub and SourceForge are often used to organize OSS development, a focal organization takes the lead in many successful OSS development projects, and many OSS developers are paid. However, much OSS research still examines individual efforts and traditional OSS products such as Linux, Apache, MySQL and PHP (a general-purpose scripting language) rather than the more focused strategic development of OSS 2.0.

---

5 For information about Gnu Emacs, see http://www.gnu.org/software/emacs/.
7 For a description of GitHub, see http://en.wikipedia.org/wiki/GitHub.
Conceptually, OSS development can be classified along three dimensions (Governance, Organizing Entity and Software Features), as depicted in Figure 3. The horizontal axis comprises the two main OSS development governance models, namely, Bazaar and Cathedral. The vertical axis indicates whether the product is developed independently by the OSS community or guided by a specific company. The diagonal axis indicates whether the product is horizontal infrastructure (e.g., platform software) or in the vertical domain (e.g., business software). The former is developed using the Bazaar model, the latter using the Cathedral model. While early OSS tended to focus on horizontal infrastructure (e.g., operating systems), OSS 2.0 tends to focus on vertical domains that require developers to have business-specific knowledge.12

South Korea’s eGovFrame Project: Openness as a Strategic Choice

Different major software vendors involved in e-Government projects tend to have their own development frameworks, and dependence on IT vendors’ frameworks raises several issues. First, the business logic of an application depends on the framework. Second, as the framework tends to be a black box, only the framework provider can maintain an application. Third, engaging with multiple vendors often results in multiple frameworks, which in turn results in redundant activities for application process setup, recruitment, education and maintenance. These issues apply to both public organizations and private companies.

The South Korean government recognized there was a clear need for a standardized development framework for its e-Government projects. However, there was significant resistance from various stakeholders. Some smaller vendors were concerned that the project for developing a standard framework would be awarded to the major software companies, resulting in them becoming dependent on the major companies and vendor lock-in. Further, as the framework would have more than three billion lines of codes, some vendors were concerned that it would be impossible for anyone else to understand the internal logic of the framework, resulting in undue advantage for some companies. Others were concerned that if the government would not pay for the framework to be developed, the companies...

involved in its development would suffer from decreased profitability. While large vendors were concerned that an OSS development framework would weaken their market dominance, smaller vendors were concerned that large companies’ participation in developing the framework would bring them an unfair advantage.

Despite the reservations, the eGovFrame project was launched in 2008, under the direction of the National Information Society Agency (NIA) and the Ministry of Public Administration and Security (MOPAS) (see text box).

Another concern was that the expected huge cost of maintaining the framework would make it difficult for the NIA to maintain it because it had insufficient staff. After several rounds of heated debate among various stakeholders, it was suggested that this concern could be mitigated if all source code and the architecture were opened to all vendors. As a result, the NIA and MOPAS examined the feasibility of openness. However, there was no precedent of government agencies opening their source code to the public. It seemed to be a risky and radical approach. MOPAS conducted a survey on the perception of an open source development framework among 483 developers (63% of whom had experience with open source projects). The results showed that software firms had encountered challenges in finding skilled developers, in the lack of technical support and in long training periods. However, they reported that the quality of the finished software was generally satisfactory.

It was decided to place the eGovFrame project under the auspices of the Department of Standards. The director of this department had previously been in charge of web services quality management standards as the chairman of the OASIS Technical Committee. Because of this experience, she strongly believed that collaboration based on standardized processes could foster innovation as well as the integration of knowledge. This belief was articulated in the form of an open innovation strategy, which encompasses open sourcing, open processes, open outputs and an open ecosystem, as shown in Figure 4.

---

### NIA, MOSPA and eGovFrame

As the steward of public-sector IT policy, the NIA acted as the project management office (PMO) in IT projects. An internal examination of e-Government projects from 2004 to 2007 indicated that proprietary development frameworks were used in 71% of projects and that vendor lock-in was a key issue. From 2007 to 2008, a management scheme for common services was established, resulting in the plan for integrating e-Government projects through standardization and expanding the use of common services. As the result, a three-year plan (2008-2010) for implementing eGovFrame and the use of common components was developed by the NIA and the Ministry of Public Administration and Security (MOPAS, now called the Ministry of Security and Public Administration, MOSPA). The implementation schedule is shown below:

<table>
<thead>
<tr>
<th>Year</th>
<th>Standard Framework</th>
<th>Common Components</th>
<th>Budget</th>
</tr>
</thead>
<tbody>
<tr>
<td>2008</td>
<td>Runtime, Development</td>
<td>142 Types</td>
<td>$3.6 million</td>
</tr>
<tr>
<td>2009</td>
<td>Development, Management</td>
<td>30 Types</td>
<td>$2.5 million</td>
</tr>
<tr>
<td>2010</td>
<td>Operations</td>
<td>47 Types</td>
<td>$3.4 million</td>
</tr>
</tbody>
</table>

The aim of eGovFrame was to improve the quality of e-Government services and the effectiveness of information and communications technology (ICT) investments. The framework, which comprises a standardized set of software tools for developing and running e-Government applications, enables developers to integrate the development environment. It also provides standards for IT managers to manage the maintenance environment and allows them to better manage their tasks using the library of e-Government-specific OSS components. The standardized development environment can also help to facilitate standardization of software architecture and design patterns of applications and, as a consequence, makes it easier both for others to understand the code of implemented software and to reuse applications.

It is important to note that the first precursor to open innovation is to implement a platform with standardized architecture and interfaces. Standardization provides guidelines and policies, which are related to centralized governance. Prior research on IT governance suggests that the major benefits of this approach include economies of scope, business agility and shared services. However, in the real world, where firms have no choice but to collaborate with other stakeholders within their collective ecosystems, centralized governance is often not workable, as

---

13 https://www.oasis-open.org/
15 All money amounts in this article are shown in U.S. dollars
even an organization with power and authority (e.g., a government agency such as the NIA) has to resort to the “soft” power of persuasion and to “win-win” propositions to overcome resistance. As a consequence, the standardized platform strategy was complemented by an open source platform, which is the second precursor for open innovation. Although some of the benefits can be realized by simply adopting a platform strategy without having an open source framework, open innovation is restricted to a smaller pool of developers. In a dynamic environment, an open source platform effectively caters to diverse requirements, and customers can choose a flexible open architecture, and adopt and enhance appropriate modules. Moreover, openness can facilitate the open innovation loop as shown in Figure 4.

**Open Sourcing**

To build trust, key stakeholders (MOPAS, the NIA, large and smaller software vendors) were involved in the eGovFrame project, and open sourcing and open sharing of knowledge were encouraged. The key aim was to ensure that the project added value to the various stakeholders. The four phases of the project are described below.

**Phase 1: Design of the Standard Framework.** To standardize eGovFrame, an environmental and functionality analysis of five major software companies’ development frameworks was carried out, and the key stakeholders were surveyed and interviewed. As a result, four environments consisting of 13 service groups and 54 service functionalities were identified:

1. The runtime environment provides the common runtime platform for running software applications.
2. The development environment provides support tools for the software development life-cycle.
3. The management environment provides support tools for the efficient maintenance of eGovFrame.
4. The operations environment provides operational tools for monitoring and operating software applications based on the runtime environment.

**Phase 2: Policy Decisions Relating to the Evaluation Process and Licensing.** The primary focus of Phase 2 was on identifying and specifying common components for government-wide
Lessons for Software Development Ecosystems: South Korea's e-Government Open Source Initiative

Figure 5: Identification of Common Components

**Principle:**
1) Reduce dependency on specific vendor
2) No conflict of interest with proprietary software

**Classification:**
1) Size of granularity
2) Business versus utility

**Criteria:**
1) High probability of repeated development
2) Reusability among government systems
3) Standards adoption

**Plan the evaluation**
- 219 shared components implementation priority

**Establish criteria**
- 67 IT implementations
- 31,114 functions
- 6 development frameworks

**Collect data (evaluate)**
- 9,019 functions

**Negotiate**
- Conference and online poll

**Construct**
- 260 common components

**Analyze data (evaluate more)**
- 5,019 functions

**Unexpected discoveries require re-evaluation**

---

usage18 (see Figure 5). To foster a government-wide and in-depth discussion about components, a \(2 \times 2\) classification matrix19 (granularity of components × degree of business centricity) was used, followed by establishing criteria for the high probability of repeated development, reusability and standards adoption. To prevent repeated development of the same functions, 67 e-Government projects between 2004 and 2007 were analyzed and 31,114 functions were reviewed. After five rounds of refinement, 260 common components were identified. In-depth discussions were conducted and any potential conflicts with proprietary software led to further discussion and re-evaluation. After several rounds of workshops, 219 common components with their corresponding categories and priorities were identified.

Phase 3 and 4: Logical Test and Physical Test. The primary focus of Phase 3 and Phase 4 was selecting the most suitable OSS relevant to the desirable architecture (Phase 1) for the development framework. Note that the development framework is a combination of groups of OSS and software developed in-house. In brief, logical tests and physical tests were conducted in accordance with the newly defined OSS evaluation process.20 Several workshops were held involving IT staff, customers (government agencies) and current and potential vendors to gather information including customer reviews, expertise reports and market research, as well as establish criteria and evaluate the data collected.

The first step identified requirements for 54 services (see Figure 6), listed all OSS relevant to each service and analyzed stakeholder views for each service group. This resulted in a list of 1,300 OSS. This list was then reduced in a two-stage screening process. First, the list was reduced to the 175 OSS that met the following criteria:

---


19 This is based on Service Reference Model of National Enterprise Architecture of Government of South Korea.

1. Software licenses with no restrictions on distribution and usage: Does the software satisfy the Lesser General Public License (LGPL), Berkeley Software Distribution (BSD), Apache or relevant license policy?

2. Constraints for integration and interfaces: Does the software support integration with the target architecture as well as legacy systems?

3. Major requirements of each service: Does the software satisfy system requirements and major stakeholders’ requirements?

These 175 OSS were then screened by qualitatively assessing each according to six criteria: functionality, reliability, usability, efficiency, maintainability, and portability. In the physical test, 85 OSS were evaluated for the basic functional and non-functional requirements that had been established in Phases 1 and 2 using an open source test bed for testing performance. The constraints for integration and for eGovFrame interfaces were also evaluated. This screening resulted in 40 OSS being selected for eGovFrame.

Open Processes
Extensive opinions from over 500 stakeholders were collected, and 20 conferences involving participants from both the public and private sectors were organized. From the inception of the eGovFrame project, large software suppliers and smaller vendors, as well as customers (government and public sector organizations), were involved in establishing the openness principle, designing the architecture and creating the blueprint. There was cross-vendor collaboration, since all participants collected and reviewed detailed functions and the pros and cons of development frameworks in the market, and this led to vendors gaining deeper insights on how to improve their own development environments because they could learn from competitors’ products. Many developers in both large companies and smaller vendors recognized that a common development framework could reduce the learning effort whenever they are involved in IT implementation projects.
Lessons for Software Development Ecosystems: South Korea’s e-Government Open Source Initiative

Open Outputs
To encourage the adoption of eGovFrame, all framework outputs (664,000 lines of code and artifacts such as UML (Unified Modeling Language) diagrams and other design documents) and 219 common components were made available to various stakeholders via the eGovFrame website, together with open license intellectual property rights (IPR). This helped to create an environment of voluntary participation among developers, vendors and government officers involved in implementing eGovFrame. In addition, free training courses were provided, and 1,236 developers were certified as eGovFrame developers. Frequent offline conferences and seminars were also held, with a total of more than 1,000 participants. An open seminar dealt not only with training dedicated to the framework, but also introduced the latest OSS trends and best practices in both the public and private sectors. The eGovFrame open license IPR allows revenue to be generated through OSS, which has led to the emergence of companies that provide paid-for complementary services for OSS.

Open Ecosystem
The eGovFrame OSS community was launched in August 2010, funded by OSS promotion initiatives in the Ministry of Knowledge Economy (MKE). Initially, the community covered 219 common components and two core modules of the standard framework (the run time environment and the development environment). While developers and users could participate in the newly launched OSS community, only “committers” could modify source code. Committers were appointed by the Supreme Council (which oversees the OSS community) and were supported and controlled by the Secretary Office, project management committee (PMC) members and the highest level committee (Supreme Council). Figure 7 summarizes the organization of the OSS community as well as the roles and responsibilities for these various groups.

After launching the open community through the eGovFrame website, the eGovFrame Center was established in November 2010. Dedicated workspaces were made available for more than 30 developers seconded from both large and smaller software companies. A developer from a large company commented that “It is meaningful to develop common OSS with other participants as we also gain technical knowledge through collaboration compared to doing research alone.” The efforts to build the eGovFrame OSS community attracted participation from students and professionals, and created an environment for collaborative sharing of knowledge. Competitions for using the eGovFrame

Figure 7: Organization of the eGovFrame Open Source Community

| Supreme Council | • Decision making on committee affair
<table>
<thead>
<tr>
<th></th>
<th>• High level consultation (including the NIA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Secretary Office</td>
<td></td>
</tr>
<tr>
<td>Project management committee member</td>
<td>• Maintain library configuration, continuous integration</td>
</tr>
<tr>
<td>(Runtime/development/common component)</td>
<td></td>
</tr>
</tbody>
</table>
| Committer | • Maintain bug management, source configuration management
| (Runtime/development/common component) | • Leading project |
| Developer group | • Bug fetch, demanded function (insert/query only)
| | • Participate in project |
| User group | • Download source code and submit feedback |

June 2013 (12:2) | MIS Quarterly Executive 101
Lessons for Software Development Ecosystems

framework were held, which helped to share best practices and to attract software expertise.

In January 2012, the eGovFrame Center appointed new committers, including a winner of one of the competitions, a leader of the biggest OSS community in Korea, and developers who had been involved in developing eGovFrame. At that time, the number of eGovFrame OSS community members had grown to more than 3,500.

At the start of the eGovFrame project, the NIA had planned to adopt the framework for five representative projects it would be managing. However, after the first phase of implementation (from about November 2008 to April 2009, see text box earlier) and the second version released in the middle of the second phase of implementation (September 2009), 17 organizations were using the standard framework for 23 IT implementation projects.

Since the NIA acts both as the PMO for e-Government projects and as the technical IT expert for the South Korean government, it can send out RFPs (requests for proposal) that require vendors to adopt eGovFrame. As a consequence, by June 2012, more than 180,000 developers had downloaded eGovFrame and used it for 227 projects, with a total budget of $701 million. These projects were in 63% of government agencies in South Korea. Officers of MOPAS commented that “This is strong evidence of how government agencies have weakened vendor lock-in and adopted the flexible framework.” The number of eGovFrame downloads is the highest ever recorded in South Korea for OSS downloads.

Impact of eGovFrame

eGovFrame enhances the relationships among government agencies, between government and the public, and between government and businesses. Specifically, eGovFrame helps to ensure interoperability, which is necessary for information sharing and linkages among government agencies, thereby improving inter-agency collaboration. Further, eGovFrame provides standardized templates and interfaces for various e-Government services. Easy implementation of linkages between e-Government systems increases public satisfaction by offering one-stop services that were unavailable previously because different government agencies offered non-integrated services. In addition, developers of e-Government services are able to improve their core competencies, technological capabilities and productivity by using eGovFrame. The standardized framework also helps to strengthen the competitiveness of smaller software vendors by offering them an equal chance of bidding for e-Government projects. Smaller vendors do not have their own individual software frameworks, and eGovFrame enables them to deliver high-quality solutions. As well as using eGovFrame for e-Government projects, smaller vendors have also used it to gain the benefits of improved software quality for projects in the private sector (such as the finance and logistics industries).

To better illustrate the impact of eGovFrame, Table 1 summarizes the situation before and after eGovFrame implementation for public-sector projects.

Private-sector companies that have used eGovFrame for non-government projects experienced significant productivity and quality improvement in system development and system operation. For example, Meritz Fire & Marine Insurance Co, Ltd. (a South Korean insurance company) used eGovFrame to improve its quality control during system development and system operation. Productivity improved by 13%, and eGovFrame enabled enhancements/upgrades to be carried out more easily. Previously, there was no development standard, errors were difficult to detect and enhancements were time consuming due to the need to analyze each program developed without adhering to a standard software framework (see Figure 2).

Lessons for Managing Development Framework Ecosystems

From our analysis of the way in which South Korea’s eGovFrame was built and implemented, we have identified six lessons for managing development framework ecosystems. For each lesson, we also provide an actionable practice.
Lessons for Software Development Ecosystems: South Korea’s e-Government Open Source Initiative

Table 1: Before and After eGovFrame in Public-sector Projects

<table>
<thead>
<tr>
<th>Before eGovFrame</th>
<th>After eGovFrame</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Redundancy of similar functions such as log-in and bulletin boards</td>
<td>• Reduced budget and enhanced productivity when implementing e-Government applications through reusable common components</td>
</tr>
<tr>
<td>• Separate e-Government systems for different government agencies</td>
<td>• Seamless e-Government</td>
</tr>
<tr>
<td>• Nine common components</td>
<td>• 219 common components with high reusability in implementing e-Government application</td>
</tr>
<tr>
<td>• Lack of application software development standards</td>
<td>• Laid the foundation for e-Government by providing a standardized development framework</td>
</tr>
<tr>
<td></td>
<td>• Provided standardization tools necessary to implement e-Government application services; by 2011, eGovFrame had been used for 192 projects</td>
</tr>
<tr>
<td>• Vendor dependency: 80% of e-Government projects are awarded to large companies</td>
<td>• Reduced vendor dependency through using eGovFrame; more than 60% of projects now awarded to smaller vendors</td>
</tr>
<tr>
<td>• Smaller vendors unable to compete as large-scale development frameworks were led by large companies</td>
<td>• Enhanced competitiveness of smaller vendors through using the advanced, specialized and free eGovFrame</td>
</tr>
<tr>
<td></td>
<td>• Improved smaller vendor capabilities through availability of a development framework optimized for e-Government implementation and by offering consulting services such as application education</td>
</tr>
</tbody>
</table>

(Source: National Information Society Agency, South Korea)

Table 2: Before and After eGovFrame in Private-sector Projects at Meritz Fire*

<table>
<thead>
<tr>
<th>System Development</th>
<th>Before eGovFrame</th>
<th>After eGovFrame</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis/design/implementation</td>
<td>No development standard</td>
<td>Quality control based on standard framework</td>
</tr>
<tr>
<td>System design</td>
<td>Lack of integrator: depended on technical architect experience</td>
<td>Quality control based on well-defined architecture</td>
</tr>
<tr>
<td>Integration test</td>
<td>Unpredictable errors</td>
<td>Proactive error control</td>
</tr>
<tr>
<td>Overall productivity improvement</td>
<td>Not applicable</td>
<td>Estimated 13%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>System Operation</th>
<th>Before eGovFrame</th>
<th>After eGovFrame</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fault tolerance</td>
<td>Hard-to-trace faults</td>
<td>Easy-to-trace faults</td>
</tr>
<tr>
<td>Enhancements/upgrades</td>
<td>Need to analyze each program</td>
<td>Easy to upgrade based on framework</td>
</tr>
</tbody>
</table>

Lessons for Software Development Ecosystems

Lesson 1: Focus on Collective Stakeholder Benefits to Secure Buy-in

When the NIA launched the eGovFrame project, it worked with analysts, designers and implementers from 11 software vendors (both large and smaller companies) to help design and build eGovFrame. This enabled it to gain as much knowledge as possible of current best practices in development frameworks. It also consulted key stakeholders from government agencies. In all, the NIA collected extensive opinions from over 500 stakeholders and ran over 20 meetings involving representatives from public and private organizations. The aim was to understand and address key issues and concerns, as well as to emphasize the collective benefits of a standardized development framework.

There were two key issues for government agencies: vendor lock-in and reusability of software. These key concerns may not be new, but they are recurrent problems that pose significant challenges. Key issues for vendors were software quality and development productivity; this is why the larger vendors had built their own development frameworks to improve productivity and reduce costs. However, these vendors reported that staff turnover and the need to train new staff on their frameworks often resulted in lower productivity and recruitment difficulties. The NIA positioned the eGovFrame project as a solution to the issues facing both government agencies and vendors.

Importantly, the eGovFrame architecture and priority of each module were decided in the information strategic planning workshop, which involved vendors and government agencies. They spent much time on developing the architecture and analyzing the environmental and functionality aspects of IT companies’ frameworks. As a result, the architecture was defined with four environments consisting of 13 service groups and 54 service functionalities. This effort helped all participants to share mental models, which prior research on the importance of a shared design process in software development had identified as being of crucial importance.21 Focusing on collective stakeholder benefits from the start of the project helped increase all stakeholders’ commitment to eGovFrame.

**Actionable Practice 1.** Development framework ecosystems must involve key stakeholders as well as emphasize the benefits of working together and sharing information throughout the ecosystem. By doing this, stakeholders are more likely to adopt collective mental models, which will help to reduce costs, avoid duplication of work and secure buy-in to the project. If the framework addresses stakeholders’ concerns and enhances their collective benefits, they will find it hard to ignore the framework and will thus be motivated to get involved with it.

Past studies have shown that owners of proprietary development frameworks co-create value with other firms in their ecosystem by encouraging complementary innovations and network effects. Independent software vendors who are part of the ecosystem often derive benefits such as increased sales because they have access to a larger pool of potential customers than if they acted alone.22

Lesson 2: Evolve from Open Sourcing to Open Outputs

To reduce the dependency on major software vendors, well-known and proven OSS were evaluated and selected for eGovFrame, and during the evaluation process, rigorous logical and physical tests were conducted. However, the experience with eGovFrame shows that openness is necessary not only for innovation, but also for cultivating trust and solving problems. Sharing source code for government software with the private sector was a radical decision. In addition, beyond sharing with private companies, all the assets (source code and modeling documents) were available on the eGovFrame website and shared with the public through the OSS community. Vendors were also free to use eGovFrame resources for private-sector projects. Vendors were happy to participate in the development of eGovFrame because they can use the framework’s source code under the free licensing policy and benefit from


selling commercial software that complements eGovFrame.

**Actionable Practice 2.** Open innovation (or management by consensus) is essential for building a development framework ecosystem. The challenge of this approach is that it is difficult to persuade participants to share their knowledge because they are concerned that the platform owner or other participants may replicate their technology and begin to offer a competing product. The conventional approach to mitigating these risks is for a software vendor to protect its IPR through patents and copyright.

Our view, however, is that for a software development platform ecosystem, open source licensing is more appropriate. Such an ecosystem requires the coordination of many fluid, organic, diverse and difficult-to-manage relationships with many different types of parties, and this coordination will be facilitated by open source licensing. By making outputs (source code) openly available, an open ecosystem is created that facilitates open innovation. For example, IBM made available the source code for its Eclipse integrated development environment (IDE) (valued at $40 million) and, as a consequence, created a huge market for complementary and proprietary products based on Eclipse. Other software firms (such as Sun with NetBeans and BEA with Beehive) have also moved their proprietary IDEs to open source.

**Lesson 3: Operate Openly but Under Strong Formal Governance**

Even though the eGovFrame source code can be shared openly, the NIA ensured that changes can be made only by authorized persons. Compared to other Bazaar-type OSS communities in which many projects are started and dropped in an ad-hoc manner, the eGovFrame community has a formal structure, with four levels of membership: project management committee, committer, developer group and user group, with oversight from the Supreme Council. Community members are appointed or promoted to a particular level by the NIA, with members at a higher level having greater discretion on how to maintain eGovFrame. Any important changes to the eGovFrame architecture and source code are subject to rigorous version-management procedures.

Additionally, a dedicated eGovFrame Center (also known as the Private-Public Cooperation Standard Framework Center) was established in 2010 to support both users and developers of OSS. For example, if a developer posted a question on the OSS open community website, staff of the eGovFrame Center answered the question in the same way that commercial vendors did on their websites. While eGovFrame project staff were drawn from both the NIA and private companies, all staff shared the role of supporting and communicating with the OSS community.

The NIA allowed eGovFrame committers (lead members) to also manage some projects in GitHub. This means that eGovFrame involves the broader open source community, numbering about 5,000 developers in South Korea. Given the fact that GitHub is the most popular OSS source code repository site and can facilitate innovative improvements to OSS products, the need for rigorous eGovFrame version management seems out of step. However, version control helps the NIA to exploit OSS innovation while having an effective governance scheme for eGovFrame.

As shown earlier in Figure 7, there is a hierarchy in the eGovFrame open source community. The Supreme Council (comprising high-level champions of eGovFrame) oversees the community and is responsible for appointing committers. In addition, any important changes in architecture and source code are reported to the Technical Standards organization, namely the Telecommunication Technology Association in South Korea, to trace important modifications and let the public know about them.

**Actionable Practice 3.** Development framework ecosystems need to expand to facilitate vibrant operation. To enable this,
such ecosystems may allow participants to integrate different collaboration networks to accelerate evolution through joint learning. Additionally, IT governance structures and mechanisms should be integrated to facilitate effective communication among participants. A proper governance mechanism, such as a steering group or hierarchy, should be put in place before participants are encouraged to expand the scope of the ecosystem. Prior research has shown that an organizational hierarchy has benefits such as the ability to maximize alignment among different activities, optimize interfaces, and reduce risk, uncertainty and variability. A formal governance structure helps to focus community attention on value co-creation and mutual benefits.

**Lesson 4: Develop Horizontal Infrastructure and Vertical Domains Concurrently**

Small but important components of eGovFrame were initially released according to priorities determined by the architecture and a three-year master plan. Subsequently, additional components were rolled out to complete the more advanced framework. Early on, only the runtime and development environments of eGovFrame were available. Even so, these were sufficient for the government agencies that would be using eGovFrame to realize that 142 kinds of sharable services were available and that adopting the framework could result in cost savings. Hence, adopters of eGovFrame viewed it favorably, and the diffusion rate of the framework tripled each year from 2009 to 2011.

Another significant factor that contributed to the rapid diffusion of eGovFrame was that the NIA divided the project into two parts—horizontal infrastructure (development framework) and vertical domain (common components for sharable services).

The development of eGovFrame also illustrates that different models of OSS development are appropriate for different purposes. The community-based Bazaar model is suitable for development of horizontal platform infrastructure (e.g., operating systems), while the company-based Cathedral model is suitable for development of OSS in vertical domains (e.g., healthcare).

**Actionable Practice 4.** The development framework and applications should be implemented concurrently. The more applications that are available on the platform, the more customers will be willing to see the framework as a source of suitable applications. While the framework itself may not mean much to customers, the availability of diverse applications makes the framework useful for customers. Participants in the framework ecosystem should therefore exploit synergies to develop both the framework and applications concurrently, as Apple has done with its App Store (platform) and Apple apps (applications). To achieve this, the framework architecture must have the flexibility to allow both platform and applications development.

**Lesson 5: Build Trust and OSS Mindset for Ecosystem Cohesiveness**

Governments are generally reluctant to open their internal decision-making processes to the public. However, the experience with eGovFrame illustrates how government's trust in citizens can establish effective collaboration between the government and the public, and help build a common mindset for ecosystem cohesiveness.

Initially, the government agencies were concerned about making the software source code available on the government website. Moreover, both private and public OSS communities in South Korea were not mature. But since usage of eGovFrame was greater than expected (as evidenced by the record number of downloads in South Korea), citizens trusted the durability of the eGovFrame project and joined the eGovFrame OSS community, thereby fostering a mindset that embraces OSS. The expanding community enabled new information and knowledge to flow to government IT projects.

**Actionable Practice 5.** Building a knowledge-sharing culture is essential for the success of a development framework ecosystem. However, cultivating trust and a willingness to share is not easy. One effective way is to publicly support the open source movement as this provides individual participants with the confidence that they are participating in a worthwhile project to build a platform that will
be used extensively. Participants will be willing to share only if they trust that their knowledge will never be plagiarized or unfairly used by competitors and if outputs are shared openly. Making source code available and allowing partners to solve the problems collectively fosters an OSS mindset. Without trust and a common OSS mindset, it will be difficult for the ecosystem to flourish.

**Lesson 6: Extend the Development Framework for Use with Emerging Technologies and in Other Countries**

The South Korean government recognizes that rapidly changing technology will impact eGovFrame. It tracks new technology developments through various industry seminars, IT awards and discussions with consulting groups and key players in the IT industry. These discussions have identified the need for a new development framework for mobile e-Government services. As a consequence, the NIA issued an RFP that requires a mobile development architecture with strong compatibility with the existing eGovFrame architecture. By doing this, the NIA is balancing stability and change by incorporating the new trend of mobile services.

Another new, and increasingly prevalent, trend is cloud computing. Based on experience with the eGovFrame project, the South Korean government’s CIO could envision the potential of cloud computing. She therefore established a private cloud center in the dedicated data center to evolve eGovFrame into an external cloud service, to be called G-cloud. G-cloud is an example of a platform-as-a-service (PaaS) that will provide a cost-effective solution for the national cloud computing development market. G-cloud is an ambitious experiment and is an attempt by the South Korean government to lead the technology trend on cloud computing with its own product.

The NIA also encourages the eGovFrame community to suggest future goals for OSS during the monthly seminars. One suggestion was the need to combat software “bloat”27 as OSS evolves. Interesting suggestions with potential impact are supported by the NIA, which calls these new projects “our wild experiments.”

As well as extending eGovFrame for use with emerging technologies, the South Korean government is promoting its use in other countries, with the support of global or regional MDBs (multilateral development banks such as the World Bank and the Asian Development Bank). The NIA recognizes that “government agencies in developing countries also have a threat of vendor lock-in (especially by foreign vendors). In this case, their fear of foreign vendors’ opportunism may be greater than with its domestic companies.”28 In addition, a key aspiration of many developing countries is to catch up with developed countries. In a short period, the NIA has adapted eGovFrame for use in nine countries, helping them to use the eGovFrame infrastructure and training their officials. Thus, eGovFrame is now recognized as a brand outside of South Korea and has the capacity to be implemented in more countries.

**Actionable Practice 6.** Once a development framework ecosystem has become established, participants should plan to evolve the framework by diversifying its services and achieving substantial economics of scope. This will not only satisfy the needs of different customers, but enable the framework to build a strong brand name. Diversification can occur in two ways. The first is through front-end systems that provide interfaces for different user environments (e.g., mobile applications and PCs). An analogous example is the unification of the Android operating system for mobile platforms with the Chrome operating system for PC platforms, both of which are based on the Linux kernel. In the words of Carolina Milanesi, lead Google analyst at Gartner Inc., "It signals the shift away from the platform per se and to building applications and an ecosystem ... the platform is mature and it needs to evolve, with the focus on the applications and the ecosystem, and the synergy with Chrome and Chromebooks and Android."29

---

27 Software bloat is where successive versions of software become perceptibly slower, use more memory or processing power, or have higher hardware requirements than the previous version whilst making only marginal improvements for users.


The second way diversification can occur is through back-end systems offering customers new functionality via various kinds of hardware assets (such as cloud computing). Existing proprietary development frameworks (e.g., Force.com) or utility software (e.g., VMware) may evolve to the PaaS model by acquiring OSS products (e.g., Salesforce.com’s acquisition of Heroku) or by making the source code of their products available. Additionally, in the future, platform ecosystems might be extended by combining hardware and software in innovative ways. Such innovations could emerge, for example, from the Google X project, the clandestine and secretive lab where Google is pursuing a list of “shoot-for-the-stars” ideas.

Note, though, that the necessary conditions for such reinvention are an OSS strategy aligned with enterprise strategy and a mindset open to experimentation.

Concluding Comments

Most organizations are reluctant to adopt OSS because they are concerned about reliability and maintenance difficulties. However, the success of South Korea’s eGovFrame project provides evidence of the ability of OSS to improve system development quality. The eGovFrame project contributed to increasing the diffusion and raising awareness of OSS in the public sector. This project illustrates the benefits that can flow from government (and companies) establishing an open source policy and increasing openness. The standardized open source development framework has reduced vendor lock-in, increased software reuse, enabled IT users to have greater bargaining power with vendors and provided standardized e-Government interfaces for customers. Moreover, the establishment of the eGovFrame community has facilitated the evolution of the framework through continuous improvement. We believe that the lessons learned from the eGovFrame project can be applied to helping other development framework ecosystems flourish and add value for their members.

30 Rooney, P. “VMware : Cloud Foundry PaaS will be the ‘Linux’ of the cloud,” ZDNet, April 11, 2012.

About the Authors

Sojung Lucia Kim
Sojung Lucia Kim (sojung.kim@gmail.com) is a principal researcher at South Korea’s NIA (National Information Society Agency). In more than 14 years with the NIA, she has been involved in numerous national projects, including Korean information infrastructure, the ICT project management office for the public sector, e-government information strategy and policy, and the Smart Work project for e-Gov. Her research interests include sustainability and e-Government. She is a member of the ISO 21500 (Project Management) Technical Committee and has collaborated with professional and academic institutions, including the OECD Council on ICT and environment, UN-ESCAP/APCICT and CGIO (Centre for Governance, Institution, and Organization) at the National University of Singapore.

Thompson S. H. Teo
Thompson S.H. Teo (bizteosh@nus.edu.sg) holds joint appointments at the National University of Singapore as an Associate Professor in the Department of Decision Sciences at the School of Business and in the Department of Information Systems at the School of Computing. His research includes sustainability, e-commerce, adoption and diffusion of IT, and strategic IT management and planning. He is Regional Editor (Asia and Pacific) for the International Journal of Information Management and is on the editorial boards of Communications of the Association for Information Systems, Information & Management, MIS Quarterly Executive and Omega. He has published more than 100 papers, co-edited four books on IT and e-commerce, and is a four-time winner of the SIM Paper Awards Competition.