STUDIA INFORMATICA 2010
Volume 31 Number 4A (92)

Christian MANGER
LG Nexera Business Solutions AG
Tomasz TREJDEROWSKI
Politechnika Śląska, Katedra Zarządzania i Informatyki
Jarosław PADUCH
Politechnika Śląska, Instytut Informatyki

# ADVANTAGES AND DISADVANTAGES OF FRAMEWORK PROGRAMMING WITH REFERENCE TO YII PHP FRAMEWORK, GIDEON .NET FRAMEWORK AND OTHER MODERN FRAMEWORKS

**Summary**. This article presents a discussion of most important advantages and disadvantages of framework programming with comparison to traditional programming. Particular references and examples of two modern web development frameworks were presented, Gideon, and Yii. Some other frameworks are also mentioned in the article alongside with their advantages and disadvantages.

**Keywords**: framework, Gideon, Yii

# WADY I ZALETY PROGRAMOWANIA W ŚRODOWISKU PROGRAMISTYCZNYM FRAMEWORK NA PRZYKŁADZIE FRAMEWORKÓW: YII PHP, GIDEON .NET I INNYCH

**Streszczenie**. Niniejszy artykuł przedstawia dyskusję na temat najważniejszych wad i zalet programowania, z wykorzystaniem frameworków, w porównaniu do tradycyjnego programowania. Przedstawiono w nim przykłady dwóch nowoczesnych frameworków, służących do tworzenia aplikacji i stron internetowych, Gideon i Yii. Również inne frameworki zostały wspomniane wraz z dyskusją nad wadami i zaletami ich stosowania.

**Słowa kluczowe**: framework, Gideon, Yii, PHP

## 1. Introduction to frameworks

In terms of computer programming, a software framework is an abstraction layer in which common code providing generic functionality can be selectively overridden or specialized by developer code, thus providing specific functionality. Frameworks are a special kind of software or coding libraries, that differ from them in that they are reusable abstractions of code wrapped in a well-defined application programming interfaces (API). They also contain some key distinguishing features that separate them from normal libraries [1].

Software frameworks can be seen as typical libraries or even normal user applications. But these distinguishing features separate them from libraries or normal user applications:

1) *inversion of control* – in framework, unlike in libraries or normal user applications, the overall program's flow of control is dictated by the framework, not by the caller,
2) *default behavior* – a framework has a default behaviour and this behaviour must actually be useful, i.e. must be process implementation-oriented or a working solution; framework can't be just a series of no-ops or pack of procedures or functions with no straightforward defined application as whole,
3) *extensibility* – a framework can be extended by the user usually by selective overriding or specialized by user code providing specific functionality,
4) *non-modifiable framework code* – the framework code, in general, is not allowed to be modified; users can extend the framework code, by writing own classes inheriting from classes introduced by framework, but can't modify framework code [1, 2] .

There are different types of software frameworks: conceptual, application, domain, platform, component, service, development, etc. [3].

Framework programming has been used in IT nearly since the beginning of history of software development. High increase of popularity of this approach of software and web development was observed in the same point of programming history as popularisation of social collaboration and open source ideology. Nowadays, we can observe a real boom in this area and new frameworks are being created nearly every month. This is supported by a common way of thinking presented by most modern developers which can be described by slogans like "not inventing a wheel again", "not forcing to open already open doors" and by using solutions ready to be used "out-of-the-box".

This situation isn't a surprise in any way. It can be observed that many developers, especially young with smaller experience, go through some kind of dazzle when starting development with frameworks and comparing working over such project with their past experience of traditional programming. Everything seems to be easier, faster and more comfortable, problems seem to solve themselves and, what is the most important, developers can avoid

many mistakes and common development errors in this way. Some of them are even honest enough to admit that they are again starting to learn an art of software development and everything they did before in so called *pre-framework era* – was some kind of luring it the darkness.

With this kind of approach and argumentation, framework programming can arise among some people into a kind of magic range but of course it isn't. Even the most advanced and powerful framework as good as any other It-related or not technology brings many disadvantages. This way framework programming, can be treated as modern, fast and agile software development technology, but shouldn't be understood as a kind of solution for every problem and the best or the only base for implementation of every project.

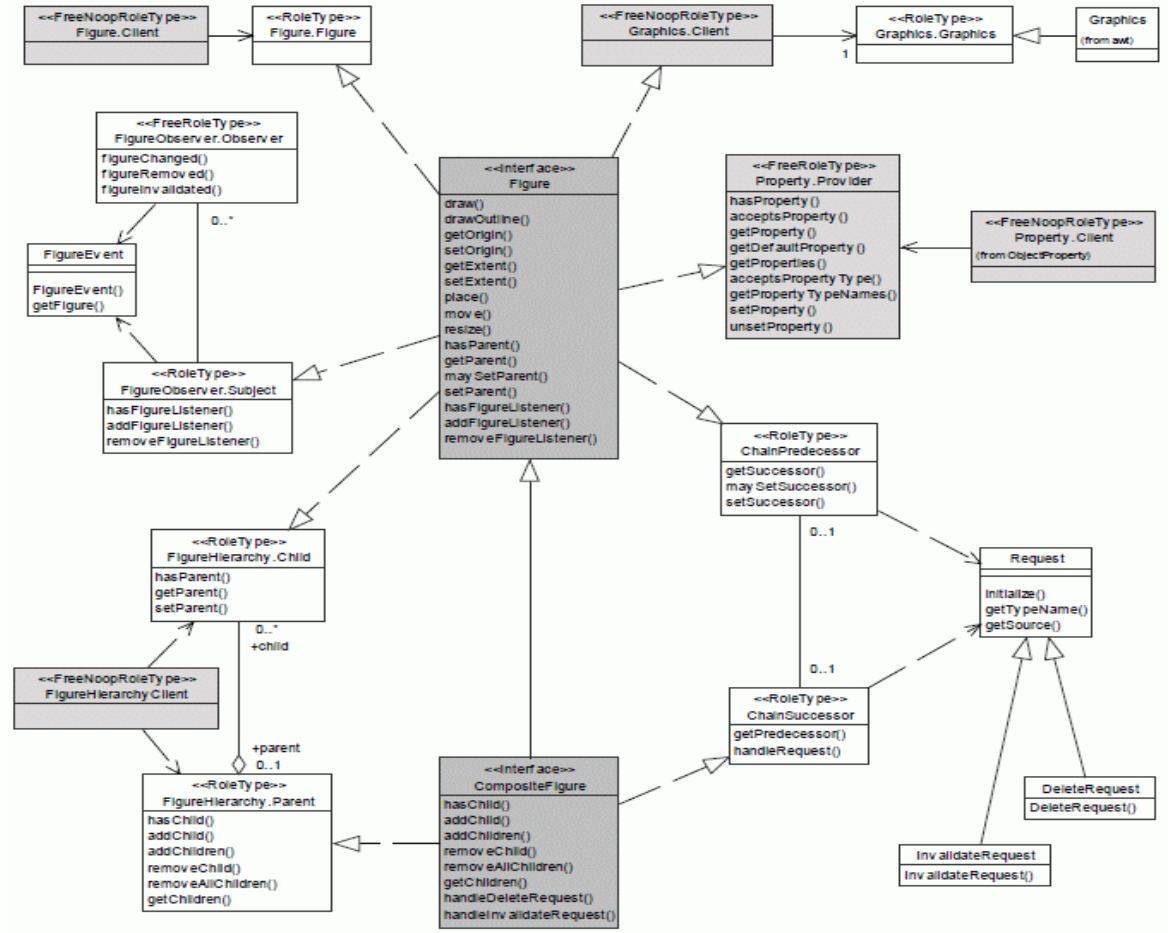



Fig. 1. Example of framework code separation into classes and a model of relations between these classes [2]

Rys. 1. Przykład stosowanego we frameworkach podziału na klasy oraz model relacji między tymi klasami [2]

One of the key features of frameworks is their complex structure focused on separating each problem solution or framework feature into separate class and strongly focusing on rela-

tions between each of these, so called *code blocks*. Example of such separation and relations has been presented on Fig. 1.

The designers of software frameworks aim to facilitate software development by allowing designers and programmers to devote their time to implementing project requirements rather than dealing with the more standard low-level details of providing a working system, thereby reducing overall development time [4]. For example, a team using a web application framework to develop a banking web site can focus on the operations of for example account withdrawals and user management rather than on the mechanics of user request handling, user authentication and authorisation processes, browser compatibility issues, etc. These, so called low-level operations are all handled by framework [2, 4].

Frameworks represent the cumulated experience of how the software architecture and its implementation for most applications should look like and knowledge of many developers for solving the most common problems. On the other hand, it leaves enough room for customization to solve some particular problems in the application [2].

Frameworks are of key importance for developing large-scale object-oriented software systems. They promise higher productivity and shorter time-to-market through design and code reuse. However, many projects report that this promise is hard to fulfil. And furthermore for some project or solutions traditional and procedural approach to development might turn out to be more efficient or less troubled in implementation despite of all advantages brought by framework programming [2].

## 2. Outline comparison

As a first step into discussion taken in this article, an outline comparison of all advantages and disadvantages of framework programming will be presented [2, 5, 7-13]. These lists present an advantages and disadvantages taken from description or user-feedback, after using many different frameworks, some dedicated for software or web-development, as good as for component, service or device programming, with no distinguish between frameworks addressed for particular implementation.

Because these are advantages and disadvantages of many different frameworks, it can be noticed that for some areas or properties, the same element (for example platform, system or browser dependency or independency) is considered as both advantage and disadvantage..

### 2.1. Advantages of framework programming

Framework programming offer a variety of advantages:

- use of code which has already been built, tested, and used by other programmers,
- no need to learn specific API nor use any low-level programming techniques, allowing to focus on business logic and implementation of project's guidelines,
- access to standardized, reusable code,
- access to large library of useful data structures and algorithms in order to manipulate them,
- all advantages of object-oriented programming,
- clean coding with using MVC (model, view controller) methodology and design patterns,
- database access abstraction layer allowing to write database-independent code and change data storing system during project development with little or no code change,
- object oriented access to database (for example DAO – Database Access Object), allowing users to manipulate on data without knowing of SQL or other data-manipulation language, treating database like an ordinary object,
- strong code modularity, allowing to program blocks of which final application is build and that can be later reused in other projects,
- ready out-of-the-box libraries, classes or solutions for common issues like user authorization or authentication, security features, database connectivity issues, caching, etc.
- platform, system or browser independency,
- noticeable application performance increase with introduction of layered caching scheme or other caching techniques,
- good prepared for easy building of front-end user interface with build-in support for templates, themes and skins as good as strong focus on logic and look separation,
- very well documented with many resources available freely in the Internet, with many books published about and often strong community behind,
- very well prepared for internationalization and localization,
- event-driven programming,
- extensible with many extensions introducing developer to many standardized protocols or solutions in many development access like authorization, verification, user management, user interface, etc.
- user-entered and database-retrieved data validation,
- error handling and logging, good testing support,

- for some frameworks (i.e. .NET) multi-language support or language independency,
- automated scripts to create an application basic skeleton, thus reducing start-time in new projects,
- simplified, standardized configuration separated in to specific file or database table, possibly with tools ready for manipulating configuration and thus allowing to centralize configuration of even large application in one place,
- for some frameworks (i.e. .NET) and under some systems (i.e. Windows), solved problem with DLL libraries access,
- for most web development framework – full URL management making developed webpages SEO friendly,

### 2.2. Disadvantages of framework programming

Framework programming offer also some disadvantages:

- platform, system or browser dependency,
- large overhead of framework code, in some situations noticeable decreasing application performance,
- bugs and security wholes discovered in framework code can (and probably will) affect every application built using it,
- for many frameworks: not suitable for development of large-scale applications or whole systems,
- sometimes a high learning curve – i.e. often require a significant education to use efficiently and correctly,
- need to follow framework coding convention, which might differ much between different frameworks, switching between frameworks may require to learn a complete new approach to coding techniques,
- hard to introduce two or more frameworks in the same project, and some frameworks may not contain all necessary libraries or classes forcing developer to write on or take an attempt to introduce more than one framework in a project,
- not supported by many popular IDE designers and in some cases (Zend Framework) framework supporting IDEs are expensive programmes, forcing developer to pay for licensee to use them,

As it can be noticed, number of advantages of framework programming strongly overtakes number of disadvantages and this is major reason why frameworks are so popular and are being chosen for implementation of very large number of modern IT projects. But, on the other hand, not so small number of disadvantages (which existence is doubted by many

framework programming zealots or fanatics) is an important element, which can be denied in discussion whether to use or not to use frameworks.

Most important advantages and disadvantages of using frameworks, taken from the table above, will be discussed in detail in following chapters.

### 2.3. Advantage: multi-target, multi-platform, multi-language (.NET framework)

Very important advantage of some frameworks is that they are not limiting developer to only one particular environment (i.e. webpage, desktop software), where his project is targeted to or to a particular device, on which this project implementation will be accessible. A good example is .NET platform developed and strongly promoted last years by Microsoft. Even if marketing name given to it is "platform", in fact this is a fully-featured framework, in the terms that it is a large library of reusable code. This definition and this kind of treating .NET platform can be found in many sources (for example [5]). This framework allows developers with little or no code fix at all to develop solutions ready to be published as desktop software applications, system services, separate self-running web pages, other web pages plug-ins and mobile devices (phone, smartphone, pad) applets. The very same code, if properly planned and organised since sketch, can be easily transformed into game-console application or even media player device (so called smart TVs and smart media players) extension application. Software and web development with frameworks this kind is truly innovative approach to software creation and extremely reduces overall development time.

For this particular framework (.NET) an even more important advantage is that it provides language Independent Multi-Language Support, thus allowing developers using different languages (Visual Basic.NET, C#.NET and J#.NET for this moment) to collaborate over the same project. This is a very innovative approaches, not available yet in many frameworks or platforms that even more reduces development time, project implementation costs and that allows to gather in one development team groups of developers using completely different languages, which before were unable to collaborate over one project or where this collaboration was by far more difficult than it is now [5].

On the other hand, .NET maybe greatest disadvantage is its nearly complete dependence on Windows-family operating systems and devices running theses systems. Nowadays we can observe large-scale activities towards platform or system independency, done by Microsoft and developers supporting .NET platform, and millions of dollars are spent each year for achieving this goal, but as a day of writing this article it still has to be admitted that .NET framework is only partially platform independent, less than more, i.e. works under Linux and MacOS X operating systems with limited portability [5, 6].

It must be pointed out here that Java programming language is not framework itself but realizes the same ideology by allowing developers to develop their application for many targets, environments or devices with little or no code change at all.

### 2.4. Advantage: Database abstraction layer and database access object

Database access abstraction layer is an advantage of many popular frameworks which allows developers to write database-independent code and change data storing system during project development with little or no code changes. This introduces an interesting flexibility for development especially of large scale applications. For example, if during long-term project programming it would turn out that currently selected relational database management system (*RDBMS*) is not able to process efficiently all requests beings sent into it (for example due to the quick growth of number of users of this project), migration to new RDBMS is a relatively easy task. And in some situations does not even require developer's attention as it does not involve code changes. It is all possible thanks to the separation of database connectivity to an abstract layer, which is supported by all mayor modern frameworks.

DAO – *Database Access Object*, and PDO – *PHP Database Object* are just two examples of database abstraction layer implementation [13].

Another important advantage of using frameworks in the area of database connectivity is an introduction of Active Record design pattern. Since it introduces object oriented access to database, it allows developers to manipulate on data without even knowing the SQL or other data-manipulation language. This is achieved by treating database like an ordinary object, where instances of class represents database tables and objects' properties relates to particular rows in table. Relations between tables are also modelled in object-related, ease to understand fashions making only really large SQL queries are not possible to be converted into Active Records design pattern approach.

### 2.5. Advantage: Internationalization and localization

There are two important elements when discussing multi-language support in application development, internationalization (*I18N*) and localization (*L10N*). Please note that a multi *spoken* language support is being discussed here, while multi *programming* language support was discussed two chapters earlier.

Internationalization (*I18N*) refers to the process of designing a software application so that it can be adapted to various languages and regions without engineering changes. For web applications, this is of particular importance because the potential users may be from worldwide. [13]. Localization (*L10N*) refers to in-application support of different date, time, num-

bers, currency etc. formatting, with respecting on local (regional) grammar and formatting issues, for example different numbering in Arabian, Roman and Cyrillic language groups.

Modern frameworks support internationalization and localization not only in the terms of ease translation of framework code, application code and supporting elements of it but also by providing ready to be use code for managing different languages, quick language change behaviour and support for URL managing classes for building SEO optimized addresses.

### 2.6. Advantage (and disadvantage): Based (or not) on design patterns

One of the most important advantages of framework programming is the fact that most modern frameworks are based on design patterns. Developers, who use frameworks, follow their coding convention which makes their code clean and extensible for future purpose [9].

Frameworks can assist in programming to design patterns and general best practices and this way helps programmers develop their projects rapidly, in agile development manner [4,9].

Fig.2. presents an example diagram of use of design patterns in implementation of one of the classes (*DrawingEditor* class) in one of frameworks (*JHotDraw* framework).

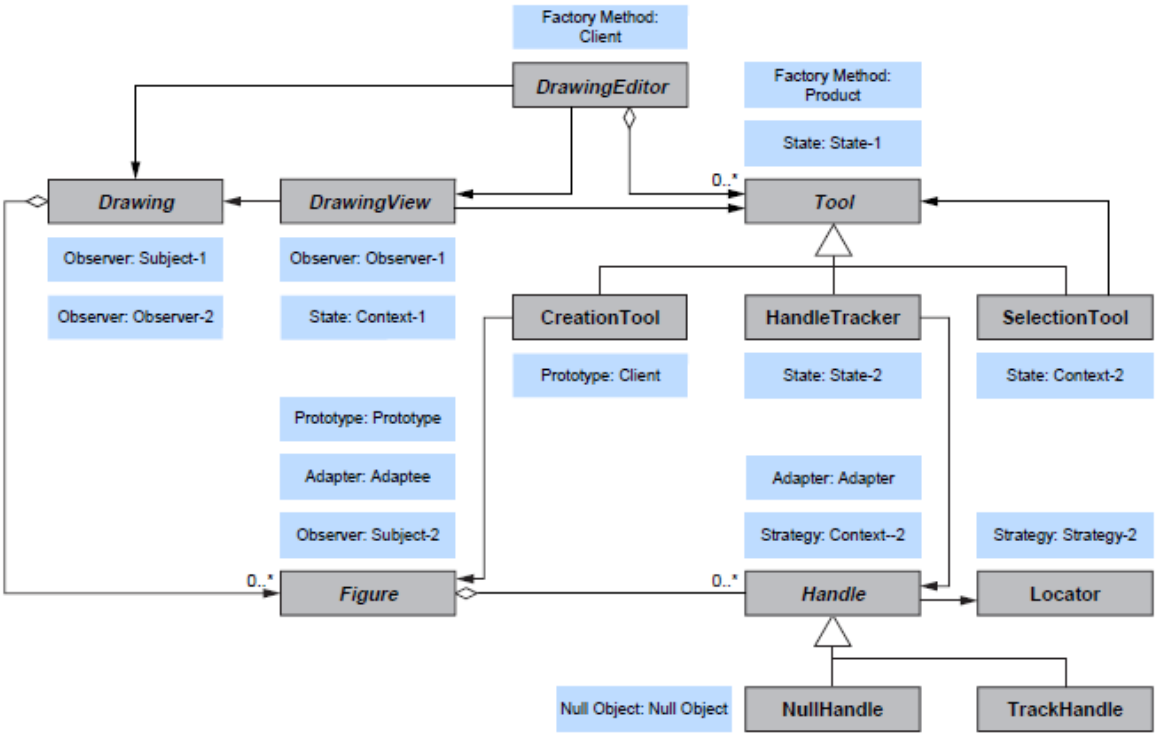



Fig. 2. Example diagram showing use of design patterns in implementation of DrawingEditor class in JHotDraw framework [2]

Rys. 2. Przykładowy diagram prezentujący użycie wzorców projektowych w implementacji klasy DrawingEditor w frameworku JHotDraw [2]

Not every popular framework implements the whole group of most common design patterns. For example, most web development frameworks based on PHP programming language implements only perhaps most famous MVC (Model, View Controller) design pattern, which separates business logic from user interface, thus making the code cleaner and extensible [9]. Some other PHP frameworks (for example Yii – see case study below) also implements Active Record design pattern for easy, agile, object-oriented database access, as it was discussed before [15].

On the other hand, since design patterns in general and MVC in particular became de facto coding standard, weak or even absent implementation of them in some frameworks can quickly become rather big disadvantage for these frameworks. For example, when speaking about one of the most popular PHP frameworks the Zend Frameworks, some developers [9] point that implementation of MVC design pattern in it is quite satisfying, while others [10] suggest that Zend's MVC feature is complete weak, view layer to simple and there is not very strong control of the front page [9, 10].

### 2.7. Disadvantage: Framework code overhead and application performance

Because every framework provides developer with a large number of reusable code libraries and classes as good as with many ready to be used out-of-the-box solutions to many common programming issues (like user authorization or authentication, security features, database connectivity issues, caching, data verification, user management, user interface, etc.) it introduces a large code overhead that is being added by default to every application, even smallest one. This can be millions of code lines or megabytes of code [10].

There are some key places in application model identified by developers where performance issues are most important and where frameworks may show out one of their biggest disadvantage:

1. Database operations using heavy-weight models – since most applications are focused over data retrieval, manipulation and processing, this can be a real harmful part.
2. Caching of pages containing many dynamic yet small elements (i.e. widgets in larger static template) – for such elements using an complex caching system may often consumer more server time and therefore reduce application performance more than not using caching at all.
3. Parsing the configuration file or database table can also negatively influence application performance especially in this parts of large-scale web applications where many fast request are processed with parsing of large configuration medium [10].

Many modern frameworks perform attempts to reduce or even eliminate negative impact on application performance with introduction of professionally designed and highly coded

caching systems and some other programming tricks. For example, Yii framework (see case study below) is using the lazy loading technique extensively, which mainly depends on theory that class shouldn't be loaded until it is used for the first time.

### 2.8. Disadvantage: Integration

Large code overhead is also an important issue when discussing integration on final customer system matters. If framework selected for implementation of particular project is a large set of classes, it might require a specific server or destination system configuration. This may introduce extraordinary cost to project funding plan.

In financial terms, integration and frameworks issues are not only related to hardware only. It also or even more are related to software and licensing issues. For example, a very big disadvantage of .NET platform is that it forces customers to run web applications developed with this framework in Windows-based server environments, which are widely known as generating far higher hardware requirements and licensee costs that their counterparts with free, open source and by far less hardware consuming Apache server and PHP language in lead.

It is highly not advisable to choose famous frameworks because of their name, fame or community, for implementation of every project as it may turn out that improper framework selection my increase financial, time or hardware resources to the levels where they can wage on whole project development [11, 12].

When discussing financial side of projects implementation and integration, it can't be forgotten that pure development process is highly related to cost-generating issues, when improper framework is chosen. Again, Apache server and PHP language frameworks win battle in this area of web-development over .NET framework and Windows-based servers as former ones offer development in may freely available, open source, yet complex IDEs (like Eclipse and NetBeans) while later requires very expensive licensee for Microsoft Visual Studio and not less expensive licensees for Windows-based server.

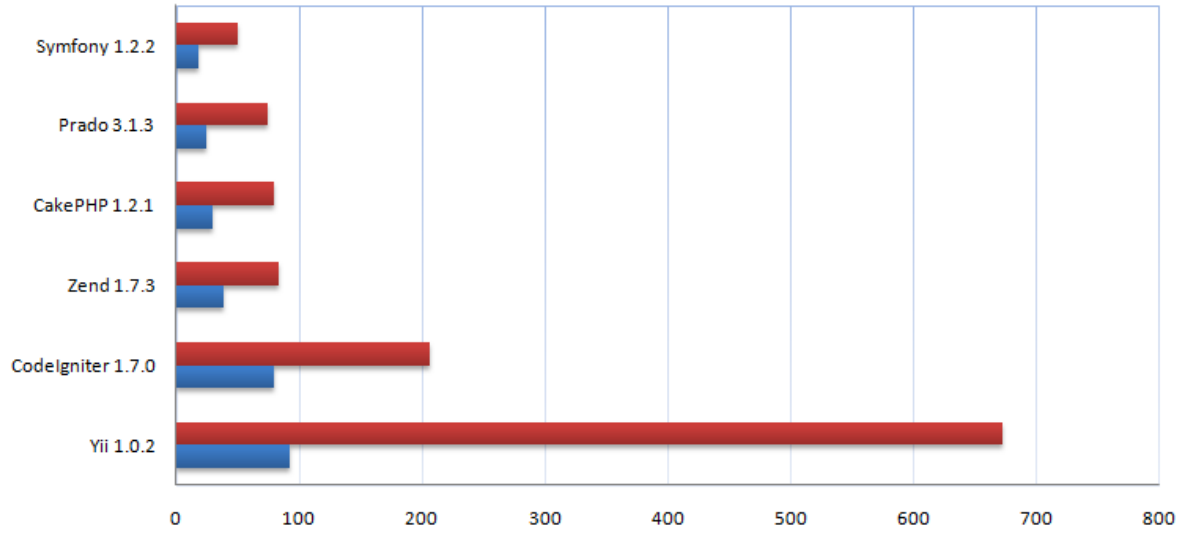
## 3. Case study: Yii

Yii PHP framework is the only framework among all most popular ones for PHP programming that fulfils all key features of modern websites and web application development, according to "PHP Frameworks" comparison webpage [13]. The only framework that also reaches this goal is Prado Framework for which Yii is an official fork. And since Prado development has been stopped in favour of Yii, this truly makes Yii the only PHP framework to

fulfil this prestigious achievement. The only area which both Prado and Yii (and some other PHP frameworks) does not support is old-concept, procedural-oriented programming with PHP 4. Since this is a very old version of PHP, dating back to 2002, not supporting it and therefore forcing developers to use modern, object-oriented programming techniques is in fact a strong advantage of these frameworks [13, 15].

It is worth noting here that among nineteen most popular PHP frameworks analysed on "PHP Frameworks" website [13], except Prado and its successor – Yii – only one other PHP framework (QPHP) supported event-driven programming technique as a date of writing this article. This coding approach, which is common, obvious and de facto standard among all desktop programming and most web-programming (i.e. .NET, JSP) frameworks is still new among PHP frameworks and support for it in Yii is another great advantage of it [13, 15].

Yii PHP framework claims to be the fastest PHP-based framework that outperforms all other in performance issue [15,16]. Fig. 3. shows related comparison.




| | Yii 1.0.2 | CodeIgniter 1.7.0 | Zend 1.7.3 | CakePHP 1.2.1 | Prado 3.1.3 | Symfony 1.2.2 |
|---|---|---|---|---|---|---|
| RPS with APC | 673 | 206 | 83 | 79 | 75 | 50 |
| RPS without APC | 93 | 79 | 38 | 29 | 24 | 18 |

Fig. 3. PHP frameworks performance comparison; RPS ("requests per second") means how many requests an application written in a framework can process per second and APC stands for Alternative PHP Cache, a caching component used for increase of application performance (in comparison to the same metering with this extension turned off) [15]

Rys. 3. Porównanie wydajności frameworków języka PHP; RPS („requests per seconds") oznacza, ile żądań wysłanych do serwera może przetworzyć przykładowa aplikacja napisana w danym Framework, a APC to skrót od „Alternative PHP Cache" – rozszerzenie języka PHP przeznaczone dla obsługi pamięci podręcznej (cache), używane do zwiększenia wydajności aplikacji (w porównaniu do tych samych pomiarów przy tym rozszerzeniu wyłączonym) [15]

According to its creators, Yii is much faster because it is using the lazy loading technique extensively. For example, it does not include a class file until the class is used for the first

time; and it does not create an object until the object is accessed for the first time. Other frameworks suffer from the performance hit because they would enable functionality (e.g. DB connection, user session) no matter it is used or not during a request [14, 15].

On the other side, a rather big disadvantage of it is that it is written in approach that does not support the most innovative, modern advantages brought to PHP development world with release of PHP 5.3 and soon – PHP version 6. Differences are so big that Yii developers decided that to fulfil PHP 5.3/6.0 needs the entire framework had to be rewritten from scratch. Due to complexity of this task, it is not scheduled to be achieved earlier than at the beginning of 2012 [13-15].

## 4. Case study: Gideon

Gideon Framework is a .NET platform framework developed and maintained by an Austrian based IT company – LGBS AG. It is not widely famous because it is mainly used for the implementation of projects requested by LGBS AG internal clients. However since it is developed in .NET platform, it is a good example of some advantages and disadvantages of web application built over this platform, not mentioned above.

Since this is mostly internal LGBS AG solution, there is no bibliography available to refer this case study. A brochure for people who want to learn framework and implementations built over it is available upon request at company headquarters.

Key disadvantages of this framework are speed and browser incompatibility. Gideon is noticeably slower than concurrent solutions built over other web framework. And since the introduction of Microsoft-like JavaScript code, not following international standards, it is also not usable at all in web browsers others than Microsoft Internet Explorer.

Gideon features fully support for drag-and-drop operations directly in browser and is open for an ease implementation of solutions that integrates different devices – i.e. projects built on the top of this framework can be accessed via standard PC with web browser, as well as through smartphone or modern cellular phone.

## 5. In general: To use or not to use object-oriented frameworks?

There are many advantages of using software frameworks in implementation of whole range of applications or projects, from simple one-case solutions to large-scale, enterprise-level system architectures. But, on the other hand, frameworks won't solve all problems and are neither suitable nor a good choice for ever project, solution or implementation. It is

a common pitfall to assume that every project will be for sure better implemented with the use of frameworks only because it uses frameworks [12].

This summary chapter will point out the most important issues of using frameworks.

Application frameworks offer a variety of advantages:

1. Using code which has already been built, tested, and used by other programmers increases reliability and reduces programming time. In a corporation this code reuse effectively saves money.
2. Thanks to framework modularization, software development teams can be split into groups developing different modules. This separation of tasks lets each team focus on more specific goals and use their individual strengths. For example, the programmers who are experts at user interface design might work on the client application while the security experts test and strengthen the framework upon which the application is built.
3. Frameworks can provide security features which are required for most of applications. This provides every application written with the framework to benefit from the added security without the extra time and cost of developing it. Examples include secure session management and escaping database input.
4. By handling "lower level" tasks frameworks can assist with code modularity. Business logic, for example, can remain in the application while the mundane tasks of database connectivity and handling user logins can be handled separately in the framework.
5. Frameworks often help enforce platform-specific best practices and rules. A desktop GUI framework, for example, may automatically build toolbars and buttons common to the local operating system. A web application framework may assist with encrypting user passwords or payment processing.
6. Frameworks can assist in programming to design patterns and general best practices. For example, many frameworks are architected according to the Model-View-Controller design pattern.
7. Upgrades to a framework can enhance application functionality without extra programming by the final application developer. If, for example, an e-commerce framework offers a new payment method, that option can automatically become available to the end user with no extra programming by the application developer [4].

There are also negative consequences of using a framework:

1. Performance can sometimes degrade when common code is used. This sometimes occurs when a framework must check for the various scenarios in which it is used to determine a path of action. It can also occur with generalized code that is not optimized for a specific situation.

2. Frameworks often require a significant education to use efficiently and correctly (i.e. some have a high learning curve). Therefore specific frameworks very often become more valuable to individual programmers when they are used repeatedly.
3. Functionality which needs to bypass or work around deficiencies in a framework can cause more programming issues than developing the full functionality in the first place. Good frameworks provide utility and structure while still leaving enough flexibility to not get in the way of the programmer. On the other hand, some frameworks are so rigid and highly structured that choosing them for an inappropriate project can be disastrous. Some frameworks are more generally suited and flexible than others. This must be carefully considered by those choosing a framework.
4. Bugs and security issues in a framework can and probably will affect every application build upon it. Therefore it must be tested and patched separately or in addition to the final software product [4].

Object-oriented frameworks promises higher productivity and shorter time-to-market of application development through design and code reuse (than is possible with traditional, procedural approaches). With using a small set of concepts (objects, classes, and their relationships), developers can model an application (analysis), define a software architecture to represent that model on a computer (design), and implement the architecture to let a computer execute the model [2, 12].

Developers who apply a framework reuse its design and implementation. They do so to solve an application problem that falls into the domain modelled by the framework. By reusing the design, application developers customize well-understood software architecture to their own specific application problem. This helps them get the key aspects of the architecture right from the beginning [2].

In contrast to non-framework based application development, a framework requires additional upfront investments. If a framework is bought (or reuses software, hardware or solution under paid licensing), it requires money to buy it and time to learn it. If a framework is developed in-house, it requires time and resources, both to develop it, and later to teach it or to learn it. However, the promise of a significant increase in productivity and reduction in time-to-market makes the investment worthwhile in most cases [2, 12].

Case studies, current practices, as well as experiences with frameworks of thousands of developers using them or collaborating in their development suggest a few key problems. Most important of them, lies in the collaboration between base classes of frameworks.

One of most important problems in construction of frameworks and disadvantages of using them, is class complexity. Classes define the behaviour of objects and their instances. Objects collaborate in multiple contexts, for multiple purposes, exhibiting task-specific be-

haviour. For complex objects, the definition of this behaviour in a single flat class interface is inadequate, and better mechanisms that describe the different aspects of objects are needed [2].

Complementary focus on classes and collaborations between them is another issue. Objects collaborate with each other, for different purposes. Much of the complexity of frameworks goes into designing and implementing the object collaboration behaviour. By assigning responsibilities to individual objects, the focus on overall collaborative behaviour is lost. Thus, mechanisms to describe collaboration behaviour are needed [2].

Third most often risen problem is object collaboration complexity. The overall collaborative behaviour of framework objects and their collaboration with client objects may become complex. To make the overall object collaboration easier to understand and to manage, it needs to be broken up into independent pieces [2].

There are also some proposition for future readings covering this subject [17-24]. The reader interested in subject discussed in this paper is invited to look at the above references for further informations.

As a final note to this article, it is important to underline that today; most large object-oriented development project uses frameworks in one way or the other. Yet, while there is no way around frameworks in software development, they are all but well understood and sometimes do not live up to their promises.

## BIBLIOGRAPHY

1. Software framework. Article at Wikipedia: http://en.wikipedia.org/wiki/Software-_framework, accessed 25 November 2010.
2. Riehle D.: Framework Design: A Role Modeling Approach. Dissertation submitted to Swiss Federal Institute of Technology, electronic version available at http://dirkriehle-.com/computer-science/research/dissertation/diss-a4.pdf, accessed 25 November 2010.
3. Shan T.: Taxonomy of Java Web Application Frameworks. Proceedings of 2006 IEEE International Conference on e-Business Engineering, electronic version available at http://portal.acm.org/citation.cfm?id=1190953, accessed 25 November 2010.
4. Framework definition at DocForge. http://docforge.com/wiki/Framework, accessed 25 November 2010.
5. .net framework advantages and disadvantages. http://www.dotnetspider.com/forum-/119309-framework-advantages-disadvantages.aspx, accessed 25 November 2010.
6. Mono (software) article at Wikipedia. http://en.wikipedia.org/wiki/Mono_%28software-%29, accessed 25 November 2010.

7. Features of Yii, http://www.yiiframework.com/features/, accessed 25 November 2010.
8. Advantages and Disadvantages of the Collection Framework. http://www.roseindia.net-/java/jdk6/collection-advantages-disadvantages.shtml, accessed 25 November 2010.
9. Advantages & Disadvantages of using different PHP MVC Frameworks. http://www-.facebook.com/topic.php?uid=114933265878&topic=14448, accessed 25 November 2010.
10. Advantages and disadvantages of several major competitions PHP Framework. http://www.softcov.com/programming-and-testing/advantages-and-disadvantages-of-several-major.html, accessed 25 November 2010.
11. Advantages n Disadvantages of Integrating Spring into Seam. http://seamframework.org-/Community/AdvantagesNDisadvantagesOfIntegratingSpringIntoSeam, published 30 July 2009, accessed 25 November 2010.
12. Java – Advantages and Disadvantages of using the Spring framework. published 30 December 2006, http://www.velocityreviews.com/forums/t389758-advantages-and-disadvantages-of-using-the-spring-framework.html, accessed 25 November 2010.
13. Website created for comparison of many web frameworks. http://phpframeworks.com/, accessed 25 November 2010.
14. Download Yii Framework. http://www.yiiframework.com/download/, accessed 25 November 2010.
15. Winesett J.: Agile Web Application Development with Yii 1.1 and PHP5. Packt Publishing, 2010.
16. Performance of Yii. http://www.yiiframework.com/performance/, accessed 25 November 2010.
17. Birrer A., Eggenschwiler T.: Frameworks in the financial engineering domain: an experience report. Springer-Verlag 1993, p. 21÷35.
18. Hill C., DeLuca C., Balaji V., Suarez M., da Silva A.: Architecture of the Earth System Modeling Framework (ESMF). Computing in Science and Engineering 2004, p. 18÷28.
19. Gachet A.: Software Frameworks for Developing Decision Support Systems, A New Component in the Classification of DSS Development Tools. Journal of Decision Systems 12 2003, p. 271÷281.
20. Pree W.: Meta Patterns-A Means For Capturing the Essentials of Reusable Object-Oriented Design. Proceedings of the 8th European Conference on Object-Oriented Programming, Springer-Verlag 1994, p. 150÷162.
21. Buschmann F.: Pattern-Oriented Software Architecture Volume 1. A System of Patterns, Chichester Wiley 1996, ISBN 0471958697.

22. Larman C.: Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process (2nd ed.), Prentice Hall 2001, ISBN 0130925691
23. Vlissides J .M., Linton M. A.: Unidraw: a framework for building domain-specific graphical editors, ACM Transactions of Information Systems 8 (3) 1990, doi:10.1145-/98188.98197, p. 237÷268.
24. Johnson R.E. : Documenting frameworks using patterns, Proceedings of the Conference on Object Oriented Programming Systems Languages and Applications, ACM Press 1992, p. 63÷76.

Recenzent: Dr inż. Damian Grzechca

Wpłynęło do Redakcji 26 listopada 2010 r.

**Omówienie**

W artykule przedstawiono dyskusję na temat najważniejszych wad i zalet programowania z wykorzystaniem frameworków w porównaniu do tradycyjnego programowania.

W podrozdziałach 2.1 i 2.2 przedstawiono najważniejsze wady i zalety tego środowiska programistycznego wraz z ich krótkim omówieniem. Następnie przedstawiono przykłady dwóch nowoczesnych frameworków służących do tworzenia aplikacji i stron internetowych. Jednym z nich jest Gideon – framework przeznaczony na platformę .NET, stworzony przez austriackie przedsiębiorstwo z branży IT, firmę LGBS AG. Drugim jest Yii, nowoczesny framework do tworzenia aplikacji i stron WWW w języku PHP, który, pomimo że istnieje od zaledwie trzech lat, gromadzi wokół siebie wielu programistów, tworzących w języku PHP z całego świata. Również inne frameworki dedykowane do tworzenia aplikacji desktopowych i stron internetowych zostały wspominanie w niniejszym artykule wraz z dyskusją nad wadami i zaletami ich stosowania. Porównanie szybkości działania różnych frameworków zostało przedstawione na rysunku 3. Programowanie z użyciem frameworków jest jedną z najpopularniejszych współczesnych metod tworzenia programów i stron WWW. Takie podejście do procesu tworzenia oprogramowania oferuje wiele zalet, wśród których najważniejsze to szybkie, zgodne ze standardami, programowanie, w niezwykle modnej ostatnio metodologii „zwinnego" programowania. Z drugiej jednak strony, programowanie z użyciem frameworków, jak każda inna technologia, niesie ze sobą pewne wady, o których nie wolno zapominać. Te wady mogą w niektórych sytuacjach stanowić podstawę przy decydowaniu, czy w imple-

mentacji konkretnego projektu powinno zostać zastosowane programowanie z użyciem frameworków, czy podejście tradycyjne.

## Addresses

Christian MANGER: LG Nexera Business Solutions AG, Kolonitzgasse 10, 1030 Wien, Austria, christian.manger@lgnexera.at.

Tomasz TREJDEROWSKI:, Politechnika Śląska, Katedra Zarządzania i Informatyki, ul. Krasińskiego 8, 40-019 Katowice, Polska, tomasz.trejderowski@polsl.pl.

Jarosław PADUCH: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-100 Gliwice, jaroslaw.paduch@polsl.pl.