e-Informatica Software Engineering Journal, Volume 4, Issue 1, 2010

Defect Inflow Prediction in Large Software Projects

Miroslaw Staron*, Wilhelm Meding**
*Department of Applied IT, Chalmers | University of Gothenburg
**Ericsson SW Research, Ericsson AB
miroslaw.staron@ituniv.se, wilhelm.meding@ericsson.com

Abstract

Performance of software projects can be improved by providing predictions of various project characteristics. The predictions warn managers with information about potential problems and provide them with the possibility to prevent or avoid problems. Large software projects are characterized by a large number of factors that impact the project performance, which makes predicting project characteristics difficult. This paper presents methods for constructing prediction models of trends in defect inflow in large software projects based on a small number of variables. We refer to these models as short-term prediction models and long-term prediction models. The short-term prediction models are used to predict the number of defects discovered in the code up to three weeks in advance, while the long-term prediction models provide the possibility of predicting the defect inflow for the whole project. The initial evaluation of these methods in a large software project at Ericsson shows that the models are sufficiently accurate and easy to deploy.

1. Introduction

Large software projects have very different dynamics than small projects; the number of factors that affects the project is much larger than for small and medium software projects. Therefore, while constructing predictions for large software projects, there is a trade-off between the prediction accuracy and the effort required to collect the data necessary to predict (designated by the number and complexity of variables). The need to collect larger number of data is particularly important as the data may be distributed over time and across the globe. Ericsson is no exception in that. The current practices for constructing predictions in large software projects at Ericsson rely heavily on expert estimations, which are rather time consuming; in particular the experts use analogy based classification techniques while constructing the predictions for defect inflow – by identifying similarities and differences between projects, the experts construct the predictions.

In this paper we present a case study conducted at Ericsson which results in developing new methods for short-term and long-term defect inflow prediction. In our case (and at Ericsson), *defect inflow* is defined the number of defects reported as a result of executing test cases during the development project at a specific timeframe, often a week, a month or a year. The term *inflow* is used in the company to denote that the defects which are discovered have to be removed before the project concludes (i.e., the product is released) and therefore these defects constitute additional work inflow in the development project.

In this paper we present two methods for predicting defect inflow in large software projects in industry:

– short-term defect inflow prediction used to predict defect inflow for periods up to 3 weeks on a weekly basis, and
– long-term defect inflow prediction for the entire project lifecycle on a monthly basis.

For each project, the long-term prediction model provides means of planning projects and allocating resources by taking into consideration expected number of defect detected which have to be removed before the release. The short-term prediction model provides the means of immediate monitoring of the defect inflow status in the project. The methods complement each other as the predictions from the short-term model need to be interpreted in the context of the predictions from the long-term model.

In this paper we also present an evaluation of these two methods in a large software project at Ericsson. The results of the evaluation show that both the short-term and long-term prediction models developed using our methods increase the prediction accuracy in comparison to the existing practices.

The structure of the paper is as follows. Section 2 presents the most relevant work for this paper. Section 3 introduces the context of the case study, in particular the organization of the large projects for which the methods are constructed. Section 4 describes the short-term prediction method, while Section 5 presents the long-term prediction method. Section 6 presents the process of evaluation of the prediction models and Section 7 presents the results from the evaluation and Section 8 validity evaluation of our study. Finally, Section 9 contains the conclusions.

2. Related Work

The most related work to our long-term prediction method is the work of Amasaki [2] who also aims at creating defect inflow profiles and trends. Their work is focused on using trends from development project to predict post-release defect inflow, which is different from our work. Our methods are intended to predict the defect inflow trends in the development project (i.e., when the software product has not been released yet) with the aim to support project management rather than maintenance of the product.

Our long-term defect inflow prediction model is similar to the the HyDEEP method [9] by Klas et al. The HyDEEP method aims to support product quality assurance similarly to our work. The HyDEEP method requires establishing a baseline for effectiveness of defect removal process which requires additional effort from the quality managers (compared to our method). The mean relative error of the predictions created by HyDEEP can be up to 29.6% which makes it a viable alternative to our long-term defect inflow prediction. In our future work we intend to compare the HyDEEP method to our long-term prediction in our industrial context to compare the complexity and ease-of-use of these two methods.

When developing the long-term defect inflow prediction model we used a similar approach to Goel [7]. Goel's process advocates for using one of the predefined defect inflow prediction model – e.g. Goel–Okumoto Nonhomogeneous Poission Model, whereas we advocate for using linear regression methods. The models considered by Goel usually consider non-linear dependencies between defect occurrences and between testing and defect discovery. Based on our discussion with experts at the company the dependencies in our case are linear cause-effect relationships.

Our research on long-term defect inflow prediction models is similar to the research on reliability theory in software engineering w.r.t the fact that we are interested in the defect inflow profile and not defect density during development. One of the most well-known models used in the reliability theory is the Rayleigh model [10], which describes the defect arrival rates for software projects after the release. It was the first model we attempted to adjust and apply before we created the method presented in this paper without much success. We use this model in the evaluation to show why this model was not appropriate for our case. Our work is substantially different from the models which predict the defect inflow after release for the following reasons:

– we can assume that when the development project is concluded, the defects reported are taken care of during the maintenance project (which can use the reliability theory to predict the defect inflow),
– changes in the trends of the defect inflow are caused by the project milestones and do not depend on the time after the release.

These underlying differences from the reliability theory underpin our research and make us have a different approach than the reliability theory.

Li et al. [11] evaluated empirically the ways of predicting field defect rates using Theil forecasting statistics. The results of their work influenced the design of our study on long-term defect inflow prediction, although we see their assumption of defects being reported after the release to be the main difference in contexts from our research. The Software Reliability Growth Model [13], which is used in their paper, seems not to be applicable to the development projects done in iterative way, since it assumes that no new functionality is developed when the defects are reported; this is not the case of the development projects.

In our future work we consider extending our prediction methods by using the same principles of Constructive Quality Modeling for Defect Density Prediction (COQUALMO) [4] and its recent extension – Dynamic COQUALMO [8]. In particular we intend to consider defect introduction and removal as two separate processes and use the efficiency of defect removal (from [9]) as one of the variables in our prediction models in the short-term predictions.

In our short-term models we consider the defect inflow to be the function of characteristics of work packages (e.g. the accumulated number of components reaching a particular milestone) and not directly the characteristics of the affected components (e.g. size or complexity). Using the characteristics of components as the sole predictors would provide us with a possibility to predict the defect density of the component and present this data on a monthly/weekly basis (based on when the component will be put under testing). Such an approach would be an extension of the current work on defect density, e.g., [3, 16, 12, 1]. In our case, nevertheless, this approach seems not feasible, because the information about how the components are to be affected by the project is not available at the time of developing predictions; in particular the change of size and complexity is not available. For short-term predictions, the data on size and complexity of components was not available on a weekly basis simply because measuring the size and complexity change is not meaningful for particular weeks; the measurements of component characteristics are done according to project plans – e.g. builds – and not on a weekly basis (i.e., not according to calendar time). In our further work we intend to evaluate if it is feasible to re-configure this data and use it as an auxiliary prediction method.

Our research on short-term defect inflow can be extended by using the research of Ostrand et al. [17] on predicting the location of defects in software components in large software systems. Predicting the location of defects can be applied as the next step after the long-term defect inflow predictions are in place, to guide the project managers into channelling testing efforts into components (or work packages which affect these components) which are historically responsible for the largest amount of defects in the system.

When developing the short-term defect inflow prediction models we considered using capture-recapture techniques [18] for estimating the number of defects in the product to assess the viability of our predictions. However, we decided to prioritize the simplicity of data collection since using capture-recapture data would require additional effort from the testers when reporting discovered defects and a more thorough statistics of the data from the defects database.

3. Context

The context of the case study is Ericsson and one of its large projects, which is developing one of the releases of a network product. In the course of development of the methods we used the pre-

vious releases of the product and we applied the methods on the new release of the product. The product has already had several releases and it can be considered as a mature one. Choosing late project releases decreases the risk of using data biased with immaturity in the organization, as it has already been shown by [21, 22] in a similar context at the same company.

As a new practice at Ericsson, the large software projects are structured into a set of work packages which are defined during the project planning phase. In the projects which we studied, the temporal aspects of defect discovery were more important than the total number of defects for the products. This was dictated by the fact that stakeholders for this particular work were project managers and at the project management level, the defect inflow is a measure of extra effort in the project (as the discovered defects have to be removed from the product before the release). The number of defects discovered at a particular point in time seems to be a function of the number of work packages reaching the testing phase. Complexity and size characteristics of the product do not have a direct impact on the number of defects discovered in a particular time frame, but the total number of defects discovered in the product.

In the existing prediction work on defect density [3, 16, 12, 14] it is usually the case that a component is developed by a single work package (or even the project, depending on the size of the component and project). In the case of Ericsson, work packages are related to the new features being developed and seldom result in creating completely new subsystems or single system components. The division of project into work packages is based on customer requirements, while the division of system into sub-systems and components is based on such elements as architectural design and the architecture of the underlying hardware (hardware/environment constraints). Each work package develops (or make changes to) components for each new large project, which makes it hard to develop a unified defect inflow prediction model using measurements at the component level. During the whole product life cycle (which spans over more than one large project – also referred to as release from the perspective of the product) the division of projects into work packages changes to a large extent (as the requirements are different for every release). Therefore using work package characteristics (which are based on distributing functionality) makes the method for long-term predictions generalizable to other projects at Ericsson.

Furthermore, from the perspective of project management, the defect inflow is also a function over the status of the project, i.e., where in the lifecycle the project is. This information is conveyed by the work package completion status. This is the assumption that we use in constructing the defect inflow predictions – that the defect inflow rate is dependent on where in the lifecycle the project is. During the project lifecycle there are 3 major milestones which are important for the long-term prediction method:

- Md – design ready milestone, which defines the point when all work-packages have finished designing. The Md milestone is used as a reference point when comparing the baseline and predicted projects,
- Mt – test ready milestone, which defines the point when all work-packages have finished their tests,
- Mf – product finished, which defines the point when the development project concludes and the product is released (the maintenance organization takes over the responsibility for the released version of the product – for that period of time the reliability theory can be used to predict the defect inflow).

Predicting the defect inflow in the large software projects at Ericsson is an important task for project planning (long-term predictions), project monitoring, and early warning mechanism (short-term predictions). Ericsson's quality managers created the predictions manually using expert opinions and analogy based techniques. The process of creating the predictions was time consuming and resulted usually in creating predictions for 2 months and interpolating the remaining months using straight lines.

The short term predictions were not widely used in the company due to the fact that they were effort intensive. The new organization of software projects provides a unique opportunity to create more accurate and less time-consuming prediction models using statistical regression techniques.

4. Short-Term Defect Inflow Prediction

In this section we present the process of developing the short-term prediction model, introduce methods used for developing the model, present the development of the model, and finally provide a short example based on a case from Ericsson.

4.1. Process

The process of creating predictions in our case started by using a baseline project data to develop the prediction models, which resulted in a set of candidate models. The applicability of these candidate models for predictions was assessed by examining the R^2 model-fit coefficient [23]. The candidate model which had the highest R^2 was chosen for further development. This model was used on a new set of data from one of the current projects to check if it was applicable for predictions. The check was done by calculating *Mean Magnitude of Relative Error* or MMRE [5]. If the MMRE was sufficiently low (below 30%, which was arbitrarily chosen by Ericsson), then the model was used for predictions. If the MMRE was higher than 30% then the "second-best" candidate model was used with the new set of data and a new MMRE was calculated.

4.2. Methods

When developing the models we used a number of statistical methods. We decided that the prediction model would be in form of a linear equation – an outcome of *multivariate linear regression* method [23]. The choice of linear regression was dictated by the dependencies between measures (*predictor variables*) at Ericsson. Based on discussions with experts in the company we could not identify polynomial or exponential dependencies in short-term predictions which dictated the use of linear model. To construct the model we used the previous release of the product, which we found to be similar in size, complexity, and maturity of project teams to the new projects in the company.

In order to avoid problems with co-linearity within our data set we used *Pricipal Component Analysis* or PCA [6]. Principal component analysis was performed prior to multivariate linear regression and was used to identify variables in the data set which can explain most variability in the data set. We did not use the principal component since our goal was to use as little data as possible and still be able to have accurate predictions.

To create the multivariate linear regression equation we used the method of Least Squares [23] for fitting coefficients in the equation and we used the model-fit coefficient R^2 [23] when evaluating whether the model can be used for predictions. When we evaluated the model on a new data set from the current project we used MMRE [5] to check how well the predictions fit the actual values of defect inflow. The evaluation methods are described in more detail in Section 7.

4.3. Model Development

4.3.1. Assumptions and Application

The short-term defect inflow prediction model was based on the following assumption: we used data from past weeks to describe the defect inflow for the current week. In other words, the dependent variable in the model was the defect inflow for the current week (which has already been known), while the independent variables were the defect inflow from previous weeks and planned/actual milestone completion status for the current week and the previous weeks.

This assumption meant that once we had the regression model describing the defect inflow for a particular week using data from the past

weeks, we could use this model to predict the defect inflow for the coming weeks by substituting data for past weeks with the data from the current week. As the predictor variables we used the defect inflow for the current project (for the time up to till the current week), and milestone completion status.

The short-term prediction model allowed predicting the defect inflow for 1–5 weeks in advance, for example: if we found that our predictor variables were number of defect inflow from 5 weeks before and the accumulated planned Md completion, using the data for the current week, we could predict what the defect inflow will be in 5 weeks. However, through simulations we found that the predictions for future weeks 4 and 5 had low prediction accuracy and therefore they are not discussed in this paper.

4.3.2. Choice of Predictor Variables

The choice of predictor variables was dictated by the availability of data and our goal – to make predictions based on the data that already existed in the organization and was easy to obtain. As discussed in Section 3, we could use milestone completion and test progress to predict defect inflow (since these were relevant variables and they influenced the defect inflow). The source of the defect inflow: testing was performed before both the design ready milestone (Md) and test ready milestone (Mt). We discovered that we needed to distinguish between Md and Mt phases as Md is only used before the Md date for the project and Mt completion status is used after the Md date.

In our case study we took into consideration the following predictor variables:

- Number of planned Md completions (prefixed with Mdp), and accumulated number of planned Md completions (prefixed with AMdp) – accumulated number of completions is the number of completions from the beginning of the project until the current week – for
  - The predicted week (Mdp_0, $AMdp_0$),
  - 1 week before (Mdp_1, $AMdp_1$), 2 weeks before (Mdp_2, $AMdp_2$), ..., 5 weeks before (Mdp_5, $AMdp_5$) the predicted week,
- Number of actual Md completions (Mda) and accumulated number of actual Md completions (AMda) for
  - 1 week before (Mda_1, $AMda_1$), 2 weeks before (Mda_2, $AMda_2$), ..., 5 weeks before (Mda_5, $AMda_5$) the predicted week;
- Number of planned Mt completions (Mtp) and accumulated number of planned Mt completions (AMtp) for
  - The predicted week (Mtp_0, $AMtp_0$),
  - 1 week before (Mtp_1, $AMtp_1$), 2 weeks before (Mtp_2, $AMtp_2$), ..., 5 weeks before (Mtp_5, $AMtp_5$) the predicted week;
- Number of actual Mt completions (Mta) and accumulated number of actual Mt completions (AMta) for
  - 1 weeks before (Mta_1, $AMta_1$), 2 weeks before (Mta_2, $AMta_2$), ..., 5 weeks before (Mta_5, $AMta_5$) the predicted week;
- Number of reported defects (defect inflow, Di) for
  - 1 week before (Di_1), 2 weeks before (Di_2), ..., 5 weeks before (Di_5) the predicted week.

To avoid problems with multi-collinearity we used the variables that were not correlated and we chose the data from the project plan which can always be obtained early in the project. A representative scatter plot for relationship between two of these variables is presented in Figure 1.

While constructing the defect inflow prediction models we deliberately did not include the data for product size/complexity as this data was not related to project planning for the following reasons:

- the software components were not assigned on a one-to-one basis to work packages and the milestones characterize work packages, not the components,
- the data on source code size was collected for milestones in the project (as it does not make sense to collect them on a weekly basis) – which means that for the whole project we could use few data points for size,
- the organization was concerned with project planning and monitoring, and not source code characteristics (e.g., size is only an in-

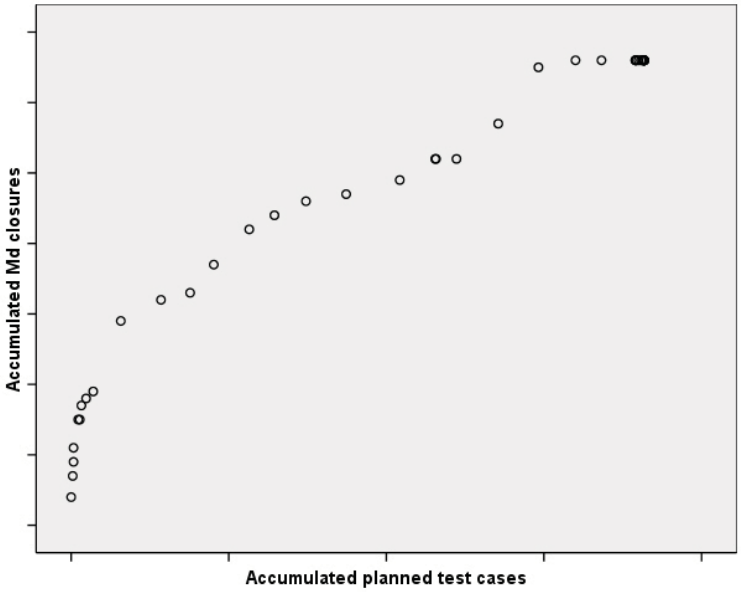

Figure 1. Relationship between accumulated planned test cases to be executed and Accumulated Md closures, Spearman's correlation coefficient: 0.96

put to the planning, but is not monitored in the projects).

Neither did we decide to use data about Md/Mt completion status as the accumulated numbers of test cases planned and executed were highly correlated with the Md/Mt completion status (Spearman's correlation coefficients [23] between 0.96 and 0.99 significant at the 0.01 significance level).

In our search for the predictor variables we used a large number of approaches and experimented with more variables than the above ones. However adding more variables did not increase the accuracy of the model significantly. For instance using all relevant variables (ca. 30) in one of the equations increased the accuracy by only 1%, but the additional effort for data collection increased significantly.

4.3.3. Reducing Number of Predictor Variables

Before constructing the regression model over the candidate variables, Principal Component Analysis was used to reduce the number of variables and thus identifying the strongest predictors. We experimented with the initial set of variables (the input to PCA) in order to achieve the best possible percentage of explaining the variability at the minimal set of measurements. PCA analysis identified 4–7 principal components (depending on the prediction period: 1, 2, 3, 4, or 5 weeks in advance). The scatter plot for the main principal components and the defect inflow is presented in Figure 2. Due to the confidentiality agreements with our industrial partner, the values on the Y-axis are not provided.

We used PCA to identify the key components and we used variables which constituted these components for the prediction models. We re-calculated the loadings in the components so that we could present the equations using the original variables and not the components (as this was one of our requirements while deploying the model in the company – to use the original names of measurements, not the names of the components).

4.4. Result: Short-Term Defect Inflow Prediction Model

The principal components before Md seemed to be linearly correlated to the defect inflow, which made the linear regression a viable technique for building the prediction model in this case. For the principal components after Md, the compo-

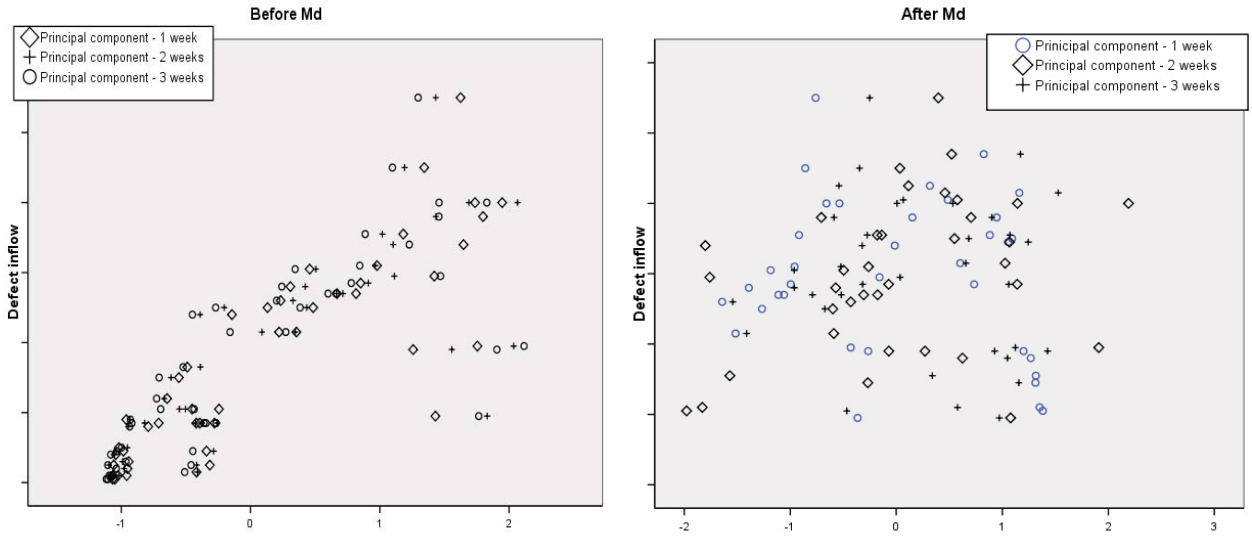

Figure 2. Scatter plot for the main principal components and defect inflow for the models before Md and after Md. The X-axes show the values of the components

nents did not show a strong correlation with the defect inflow, which affects the fitness of the regression models and the prediction accuracy. We checked whether the principal components expose logarithmic and polynomial relationship to the defect inflow, but the results showed almost a complete lack of relationship for the logarithmic curve and large scatter for the polynomial curve. This supported the claims from Ericsson experts that the relationships are linear and not polynomial or exponential. The variability of the data set explained by the principal components is presented in the last column in Table 1.

The equations used to predict the short-term defect inflow are presented together with the R^2 coefficient for the regression model and the variability explained by the component which contained the variables used in the equation. The variables used in the equations are subsets of variables presented earlier in this section.

4.5. Example

As an example, let us predict the defect inflow for a particular week in an example project. Let us assume that we are in week 10 of the project, and it is before the design ready milestone (Md). The data for that particular week is presented in Table 2. We predict week 11, so the values for 1 week before the predicted week are the values for week 10 (the current week), the values for 2 weeks before the predicted week are the values for week 9, etc. We substitute the appropriate numbers for the equations presented in Table 1 thus obtaining the predictions for week 11. By using the data from week 10 as the data for 2 weeks before, data from week 9 as the data for 3 weeks before, we can create predictions for week 12 – for example AMdp2 in equation is AMdp1 from Table 2 (because the data shows the values relative to week 10, not week 12. The value of the defect inflow for week 13 is obtained in the same way.

The results for the short-term predictions are presented in Figure 3.

The figure shows that given the current circumstances of the project (i.e., the number of defects reported in the current week and the status of the planned and accumulated numbers of work packages reaching the Md milestone) week 13 seems to be the week when the project manager should pay more attention to as the number of defects discovered is going to be rather high. The potential action of the project manager is to prepare more development resources for that week to repair the defects discovered.

Table 1. Defect inflow prediction models (short-term)

Before/after Md (design ready milestone)	Period	Equation	R2	Variability explained by component
Before Md	1 week	$D = 1.499 + 0.584 * AMdp_0 + 0.650 * Di_1 - 1.285 * AMdp_1 + 1.102 * AMda_1$	0.86	93.84%
Before Md	2 weeks	$D = 2.639 + 1.173 * AMdp_0 - 2.029 * AMdp_2 + 1.724 * AMda_2 + 0.461 * Di_2 - 0.366 * Di_3$	0.82	91.43%
Before Md	3 weeks	$D = 4.187 - 0.357 * Di_3 + 1.928 * AMdp_5 - 1.192 * AMdp_0$	0.62	90.92%
After Md	1 week	$D = 39.155 + 0.470 * Di_1 + 1.290 * AMtp_0 - 1.185 * AMtp_1 - 1.214 * AMta_1 + 0.039 * AMtp_2 - 0.044 * AMta_2 + 1.297 * AMtp_3 - 0.517 * AMta_3 + 0.003 * AMtp_4 + 0.107 * AMta_4 + 0.148 * Di_2 - 0.237 * Di_3 - 0.194 * Di_4 + 0.180 * Di_5$	0.67	65.74%
After Md	2 weeks	$D = 53.669 + 1.419 * AMtp_0 - 0.682 * AMtp_1 + 0.099 * AMtp_2 - 1.844 * AMta_2 + 0.429 * AMtp_3 - 0.768 * AMta_3 + 0.646 * AMtp_4 + 0.446 * AMta_4 + 0.306 * Di_2 - 0.265 * Di_3$	0.55	78.69%
After Md	3 weeks	$D = 24.82 + 0.47 * Di_3 - 0.351 * AMtp_2 + 0.308 * Di_5$	0.62	59.09%

Table 2. Data for week 10 of the project

Variable	$AMdp_0 - week11$	$AMdp_0 - week12$	$AMdp_0 - week13$	$AMdp_1$	$AMdp_2$	$AMdp_5$	$AMda_1$	$AMda_2$	Di_1	Di_2	Di_3
Value	5	5	5	13	14	20	12	12	4	5	5

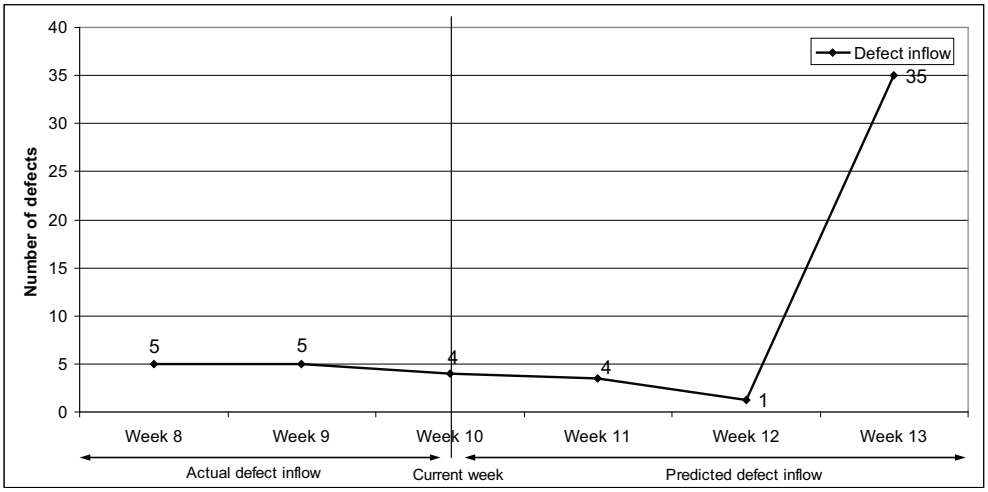

Figure 3. Short-term defect inflow prediction for week 10

5. Long-Term Defect Inflow Prediction

While the rationale behind the short-term prediction was to predict the status of the project, the rationale for the long-term prediction was to create a monthly trend of defect inflows in the project for the whole duration of the project. The process for creating the model was the same as for the short-term prediction, but the methods and results were different.

In the case of short-term models it was the equations which were the resulting model. In the case of long-term prediction it was the method which was the result. This method consists of the following steps:

1. Identify a baseline project,
2. Partition the defect inflow curve of the baseline project,
3. Create regression models for each part of the curve,
4. Calculate scaling factor for the new project,
5. Plot the defect inflow curve for the new project.

On the contrary to the short-term defect inflow prediction, the long-term prediction used a single predictor variable – project progress – unlike several variables in the short-term prediction models.

5.1. Long-Term Defect Inflow Prediction Method Development

5.1.1. Methods

For creating the regression models we used the polynomial regression method and we used only one variable – the project progress – as the predictor variable. The regression use the Least Squared Estimators similar to the multivariate linear regression used for short-term predictions.

For creating the scaling factors we used an average ratio between the number of defects reported in the new project compared to the baseline project. The calculation of this factor Sc was done using the following formula

$$Sc = \frac{1}{n}\sum_{i=1}^{n}\frac{p_i - a_i}{p_i},$$

where: n – the number of months for which we have the actual data, p_i – the value obtained by from the equation before scaling, a_i – the value of the actual data of defect inflow for this month. The rationale behind this formula was that it was an average relative difference between the predicted defect inflow and the actual data from the predicted project.

The scaling factor Sc was calculated for the curve for which there was some initial defect inflow data available. If the data were not available the scaling factor can be expert estimations (e.g. before the project start).

The method for calculating the scaling constant $Sc1$ for the curves which did not start at the beginning of the project was

$$Sc1 = \frac{pred_{ActualProject}}{pred_{BaselineProject}},$$

where: $pred_{Actualproject}$ – the predicted defect inflow for the current project for the last month for which the 1st equation (curve) can be used, and $pred_{Baselineproject}$ – the predicted defect inflow for the equation describing the second curve.

5.1.2. Assumptions and Application

The assumption for the long-term defect inflow prediction in the software project was that there existed a number of projects which could be used as a baseline. With that respect the long-term defect inflow prediction was similar to the analogy-based estimations.

The long-term prediction method should be used at the beginning of new projects in order to estimate how many defects can be discovered and when during the project. This information is particularly important for project planning and monitoring and the stakeholders for these predictions are project managers in large software projects. The long-term defect inflow prediction is usually not applicable for small projects since these usually different significantly from each

other, even if they are executed by the same organization and on the same product.

Our long-term defect inflow prediction model was designed to work best for projects already in progress as it used the actual reported defect inflow from the projects to adjust the prediction models and thus increase their accuracy. Using expert estimations at the beginning of the project should be replaced as soon as real data is available in the new project.

5.2. Result: Method for Constructing Long-Term Defect Inflow Prediction Models

It should be noted that the resulting long-term prediction model should be adjusted by the owner of the prediction in order to increase its accuracy. In particular, the defect inflow curve needs to be maintained once a month so that the predicted defect inflow is as accurate as possible.

5.2.1. Identify Similar Projects

The identification of the most similar project needs to be done by experts, e.g., the experts involved in creating the predictions. The factors that should be taken into account while identifying the most common projects are:

1. Estimated size of the project, measured as number of person-hours in the project.
2. Number of "heavy" features in the project – i.e., features which are of high complexity.
3. Complexity of the complete project – i.e., including integration complexity.
4. Time span of the project.

The most important factor is the estimated complexity. A rule of thumb used by experts is that for a project which is twice as big as a reference project, the number of defect inflow in each month would be around 75% more than in the reference project.

The time span is important only in the case when the projects are significantly longer, otherwise the method compensates for that by using the time relative to the Md milestone. Hence, similar to the short-term defect inflow prediction the Md milestone datse is an important reference point. In case the time span of the project is significantly longer, the scaling factor (described later in this section) needs to be obtained through expert estimates and not using the method described in this paper.

5.2.2. Partition the Defect Inflow Curve of the Baseline Project

In order to identify curves, identify the points where the curves change shape. These changes are important to identify otherwise we assume that the fitted equations will under-/over- predict the values of the peaks in the defect inflow. The peaks, however, are the most crucial elements to predict since project management is interested in the information how many defect might come in the peak time.

It should be noted that projects are usually independent from the calendar time – which means that the prediction model should be done according to the project milestones – e.g. Md. However, changes in the time scale should be done after the equations are built and when the models are to be applied for the current project.

5.2.3. Create Regression Models for Each Part of the Curve

The next step is to create equations describing the shape of the curves in this chart using regression methods. The equations for the curves can be identified using curve estimations methods in statistical packages. These curve estimations use the standard regression techniques – e.g. least square estimators.

In order to create the equations describing each curve (identified in the previous section) we need to:

- Use separate equation for each curve,
- Start with a straight line, and
- Change the curve type to polynomial with order set to 2, 3 or 4. The order depends on

the R^2 – as a rule of thumb, it should be as small as possible. The higher the degree of the polynomial, the higher the risk of errors in predictions when the time-scale of the predicted project is different than the time-plan of the baseline project.

These equations "re-create" the defect inflow for the baseline project using mathematical equations, which allows us to adapt the defect inflow trends for different time scales.

5.2.4. Calculate Scaling Factor for the New Project

Up to this point, the method constructs a set of equations which describe the trend of defect inflow in the baseline project. In order to predict the defect inflow in the future projects, the equations need to be scaled (moved up or down the y-axis) to reflect the actual values so far of the predicted project. The goal of this step is to have a prediction model which is in form of a set of equations which are scaled according to certain criteria. For the different parts of the curves identified so far we use different scaling factors. For the first curve, in our case there is a single criterion: the predicted defect inflow should fit the actual defect inflow from the predicted project. The outcome of the scaling is the scaling factor Sc, which is used in the following way

$$D_m(month) = Sc * y(month),$$

where: $D_m(month)$ – defect inflow for a specific month, $y(month)$ – the predicted defect inflow for the month calculated from the equations describing the baseline project.

As mentioned in the methods section there are two different ways of creating the scaling factor, which are used (i) at the beginning of the project (when no defects have been reported), and (ii) during the project when some defects have been reported. In the first case (i), the scaling needs to be done using expert opinion – i.e., the expert has to provide the ratio of complexity between the predicted and this ratio becomes the scaling factor Sc.

In the latter case (ii), the scaling can be done by "fitting" the predictions to the existing trend in defect inflow for the predicted project. Using the actual data means that we do not rely on subjective estimates but we actually try to answer the question: "How will the defect inflow look like in the predicted project if we continue with the current trend of defect inflow?" Furthermore, the predictions of defect inflow become more important once the project progresses – i.e., since the defect inflow is not expected to be large at the beginning of the project. This fitting can be done in the following ways: (i) Calculating the average relative difference, or (ii) Using non-linear regression. The first one (i) works well for the projects which has similar life span (i.e., differences in life spans should not be more than 2 months). The second (ii) is more robust and does not have this limitation.

5.3. Example: Long-Term Defect Inflow Prediction Model

In this section we present how we constructed a prediction model for one of the projects at Ericsson. In our study in one of the product the trends in the defect inflow for some of the releases are presented in Figure 4. The trends are similar, but not the same, which requires more advanced methods for predicting the defect inflow than only the analogy based estimations. Due to the confidentiality agreement with our industrial partner, we present the data scaled to the largest defect inflow in each project and we present only the subset of months for the project.

The figure shows that the trends in the defect inflow are rather stable over releases (although they differ in the values, which cannot be shown in the figure due to confidentiality agreements). They are presented in a relative time scale with the common point of Md milestone; the milestone when all work packages have finished designing.

An important observation is that all three projects have a similar percentage of defects in-

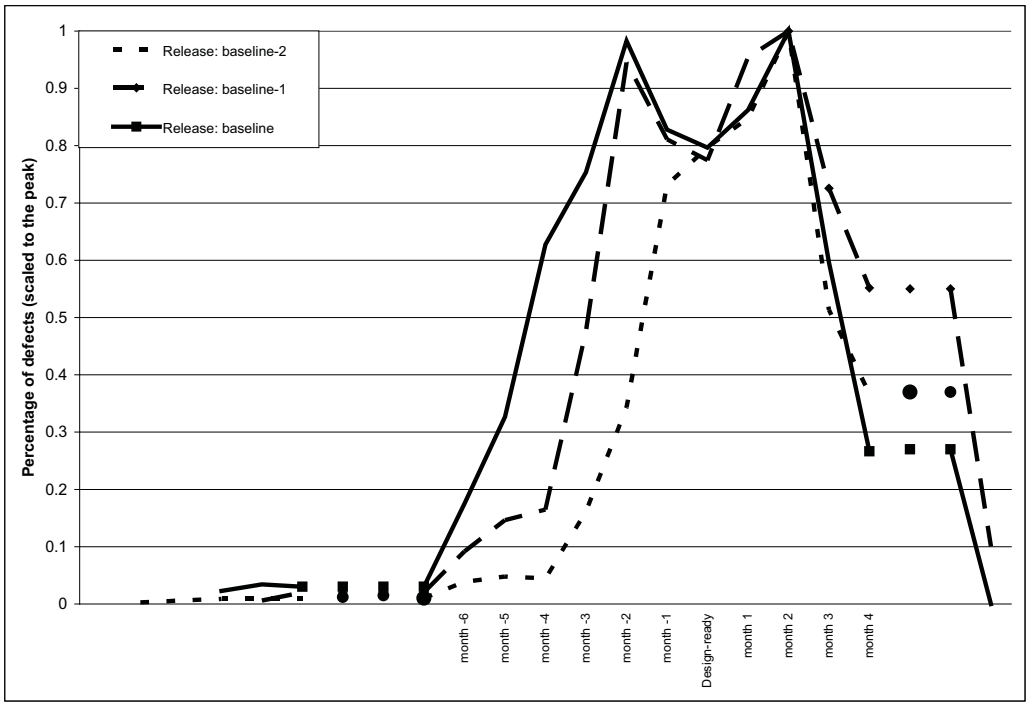

Figure 4. Defect inflow trends for previous projects scaled towards design ready milestone. The dots represent removing a number of months since we cannot show the complete time frame for project

flow at the Md milestone month. Since this is the case, we use the Md milestone month as a reference point in constructing the long-term predictions.

In order to identify the baseline project we asked experts who work with these baseline projects. They identified previous release of the same product as the best baseline ("Release: baseline" in Fig. 4).

In the case of this baseline project we identified the following curves:

1. months 1–5,
2. months 5–9,
3. months 9–11.

The developed trend line for months 1–5 is presented in Figure 5. The values of the defect inflow are normalized, due to the confidentiality agreement with our industrial partner. The curve equation is displayed in the chart, where x denotes the month. The scaling factor was calculated to be 1.23 in month number 3, which meant that the new project produced ca. 23% more defects thanAcknowledgments the previous project. We validated that with the quality managers for that project who confirmed that this number reflected their expert opinion.

6. Evaluation of Defect Inflow Models in the Context of Ericsson

6.1. Design of Evaluation

When developing the prediction models we used the model fit coefficient (R^2) to observe whether models are accurate w.r.t. the past projects used to build the models. In this section we describe how we used the data from new (current) projects to check whether the models accurately predict defect inflows in new projects. We evaluated two aspects: the ability to predict the correct value and whether predictions over a period of time (e.g., predictions 3, 2, and 1 weeks in advance) predict the same value (*stability*). Unstable models change rapidly over time which makes them less trustworthy – how can we trust a prediction model that will change a lot in the coming weeks/months?

We used the Magnitude of Relative Error (MRE) metric to measure how accurate the predictions are. MRE was defined in [5] as

$$MRE = \frac{|a_i - p_i|}{a_i},$$

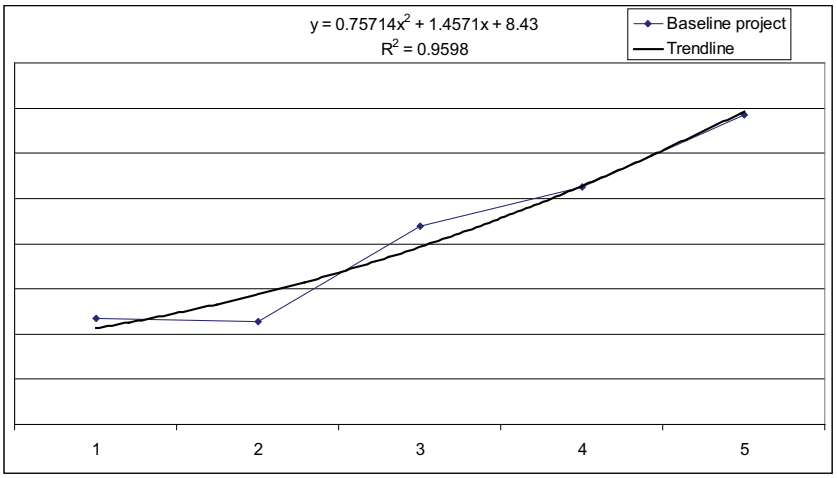

Figure 5. Equation for the curve for months 1–11

where a_i means the actual value for the defect inflow for i-th week (short-term predictions) or month (long-term predictions); p_i denotes the predicted value of defect inflow for the i-th week or month. In the evaluation we used both the distribution of MRE and mean MRE (MMRE). The best models were expected to have the lowest value of MMRE – i.e., the mis-predictions of models are small.

For evaluating the stability we used our own measure *mean in-stability (MiST)* as a metric for comparison, which we defined as

$$MiST = \frac{1}{n}\sum_{i=1}^{n}\frac{|m0_i - mj_i|}{m0_i},$$

where: n – the number of months used in the prediction, $m0_i$ – the predicted defect inflow for the i-th month created before the project (0th month), mj_i – the predicted defect inflow for the i-th month created in the j-th month.

In evaluation of the prediction accuracy we compared models developed in our research (presented in Table 1) and "average" models, i.e., predicting using a simple average amount of defect inflow in a baseline project (or the average number of defects in the current project – up to the week for which the prediction was made), and the moving averages. The rationale behind the average models was that if we did not know how to predict the number of defect inflow in a particular week, we could take the average number of defects for all weeks as an estimator; alternatively we could also use the median (i.e., the most common value of the defect inflow). Thus, in the evaluation (Figure 6), we used the following models:

- Average number of defect inflow from the baseline project,
- Average number of defect inflow from the actual project until the week of prediction,
- Moving average (2 weeks) of the number of defect inflow from the current project (i.e., the predicted value of the defect inflow is the average of the defect inflow from previous 2 weeks),
- Moving average (3 weeks) of the number of defect inflow from the current project (i.e., the predicted value of the defect inflow is the average of the defect inflow from previous 3 weeks),
- Value of the mode of the defect inflow from the baseline project (to some extend this is the use of analogy based estimation),
- Value of the mode of the defect inflow from the current project,
- Expert estimations for 1, 2, and 3 weeks.

In order to evaluate the long-term prediction models developed using this method, we compared the prediction model developed using our method to the following prediction models:

- Linear, quadratic, cubic, 4th degree, 5th degree, and exponential curve depicting the trend in defect inflow,
- Rayleigh model,

– Expert estimations, which are based on the predictions done for previous projects, prior to our research.

The above models were chosen since they require a similar amount of effort to create compared to our models. We have deliberately excluded methods like Bayesian Belief Networks [15] as their construction requires significantly more effort than the construction of the simple prediction models using our method.

6.2. Results of Evaluation

6.2.1. Short-Term Predictions

In this section we show how the models worked in a new project at the company. The new project which we chosen for the evaluation is the next release of the same product, while the baseline project was the previous release of the same product.

The values for the MMRE for the reference prediction models and the short-term prediction models are presented in Figure 6. Figure 6 indicates that the best model is the moving average for 2 weeks, which is one of the simplest models to construct. Our prediction models have larger mis-predictions, which are caused by the fact that the predictions after Md are based on the data which was weakly correlated with the defect inflow. Despite a low value of MMRE for predictions using moving averages, there is a disadvantage of these two models. Using moving averages shows trends in the defect inflow for more than one week, which are rather stable (the moving average are partially used by experts in estimating the defect inflow). However, the moving averages do not allow predicting peaks (as the peak shown in Figure 3 for week 13).

The worst prediction models are the models using modes from current and baseline projects; these two models mis-predict the defect inflow in most cases by more than 100%.The predictions made by the expert had the most common mis-predictions of 75%-90%. The predictions created using moving averages can result in less accurate models (since they have a larger percentage of mis-predictions above 90% than 'our' models). From the experiments with the historical data we found that the prediction models developed in this paper had a tendency of over-predicting (i.e., predicting values that were larger than the actual values), in particular indicating the potential "red-alerts" for the projects – i.e., showing that there will be a high raise in the defect inflow in the project. Although this might be a problem from the statistical perspective (low accuracy), this provides a means for project managers to get early warnings of potential problems so that they can have a time for reacting to some extent. This, however, cannot be verified on the historical data and we are currently in the process of evaluating the models in other large projects at Ericsson. The interview with the expert provided the predicted values for 10 different weeks. The expert was asked to make the predictions for 10 different weeks in the same way as he does the predictions when they are needed (using the information only up to the predicted week), although making short-term predictions is not done very often; the experts are focused on long-term predictions for the whole project. The results show that the methods presented in this paper are not worse than the predictions made by the expert and hence can be used as a surrogate for expert predictions. A disadvantage of the expert predictions is that they require a very deep, insight knowledge into the project. The expert making the estimations is very experienced. The time required to create the predictions was negligibly longer than the time required when using the prediction models.

6.2.2. Long-Term Predictions

The values of Mean Magnitude of Relative Errors (MMRE) are presented in Figure 7. The MMRE is calculated using 7 months after the project start (not for the whole project, since it is not yet concluded).

This shows that the best model is using a single, exponential equation. However, by examining only the first 7 months can be misleading,

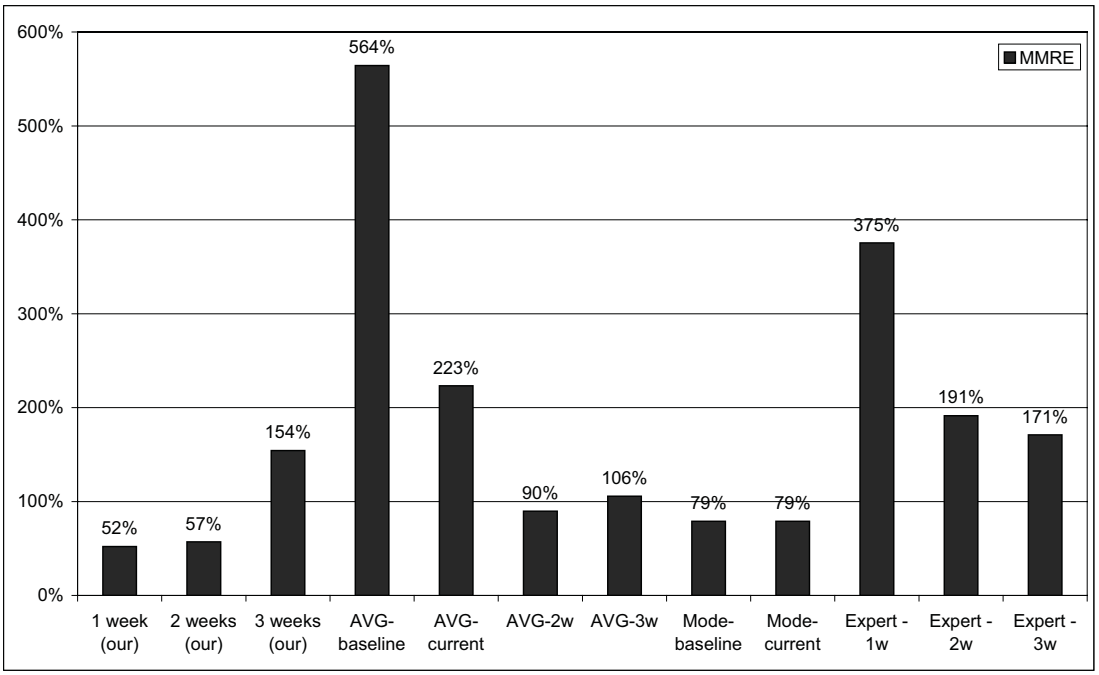

Figure 6. MMRE for short-term predictions

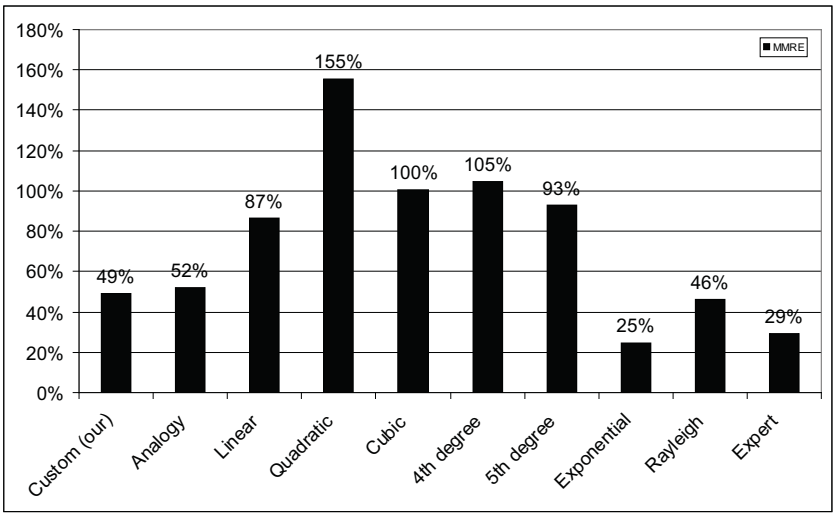

Figure 7. MMRE for the new project

as the predictions for the whole project using exponential equations do not produce trustworthy results by the end of the project. Figure 8 presents the predictions for the whole project.

The results from the stability evaluation are presented in Figure 9. A good model is not changing very much from month to month meaning that the initial predictions are actually trustworthy.

The stability need to be assessed together with the actual defect inflow trend in the project, which is presented in Figure 8. The trend in defect inflow is different that was expected, and it is different than in the baseline project. These differences caused the instability of the model. However, as the project progresses, the peak in month (a+3) was leveraged and the trend in month (b) came back to normal. How-

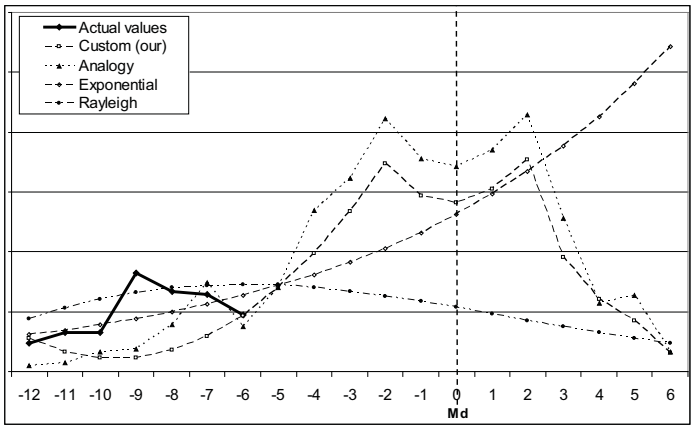

Figure 8. Long-term predictions for the new project

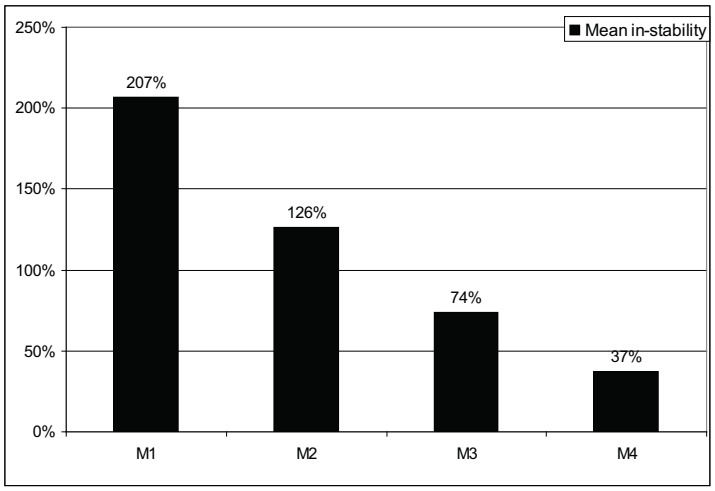

Figure 9. Stability evaluation

ever, given the history and the way in which the scaling factor is calculated, the current predictions are that the defect inflow in the project will be smaller than initially expected.

By observing the MiST chart in Figure 9 we could conclude that the predictions are stable only when there are no peaks in the defect inflow. The peaks are exceptional situations in the projects and they render the predictions unusable, thus call for adjusting the predictions. The presence of such a peak means that the prediction model should be constructed differently – e.g. by choosing a different baseline project, or splitting the baseline project into more curves (c.f. Section 5) – basically the instability is caused by the fact that we use inappropriate curves, which was caused by the fact that this peak was not present in the baseline project.

7. Validity Evaluation

In this section we evaluate the validity of our studies for constructing and evaluation of the methods. We use the framework of Wohlin et al. [24].

The main threat to the *external validity* of our results is the fact that we developed and used prediction methods in a single organization within Ericsson. Even though the organization is a large one (c.f. [20]) and we tested that at more than one product, it could still be seen as a threat. In order to minimize the threat we used the predictions in more than one project and product. The results were sufficiently accurate for both products and they also led to taking immediate actions by project managers in order to prevent potential predicted problems.

The main threat to the *construct validity* is the use of data from manual reporting to construct the prediction models. Although this might be seen as a problem issue, from our discussions with experts it was clear that the simplicity to create the predictions was one of the top priorities and hence our decision. Using test-progress data for predicting defect inflow in the same organization can be found in [19]. In order to minimize the threat that we use "random" variables without empirical causal relationships we preformed a short workshop with Ericsson experts. The outcome of that workshop was that there is empirical causality between the predictors and predicted variables.

Another threat to the construct validity is the use of regression method in our research. Using regression algorithms can be burdened with the problem of overfitting, i.e., fitting the regression equation in short term predictions (or curve for long-term predictions) too closely to the baseline project. Overfitting can cause the models to be inapplicable for other projects. We minimize this threat by evaluating our results in new projects and check the applicability of the models.

The main threat to the *internal validity* of the study is the completeness of the data and mortality of data points. It is rather a common situation in industry that data might be missing due to external factors (e.g. vacations, sick-leaves). In our case the missing data was handled by removing data points which were incomplete and removing the data points which could potentially be affected by low-quality data (mainly during vacation periods). The number of data points removed was small compared to the data sets (less than 5% of data points).

Finally, we have not discovered any threats to the *conclusion validity* as we used established statistical methods to develop the models and confirmed our findings with expert knowledge in the company.

8. Conclusions

This paper presented two complementary methods for predicting defect inflow in large software projects: short-term and long-term defect inflow prediction. The methods are used for the purpose of project planning and monitoring at Ericsson. The goal of introducing new methods in this paper is to provide the experts with support for creating the prediction models using statistical methods based on the data which is already collected in the organization (or which can be collected at reasonable costs). Using linear regression methods resulted in simple and high-cost efficient methods, which could be seen as a trade-off between prediction accuracy and costs of predicting. In this paper we tried to address this trade-off by minimizing the number of measurements to be collected and focus on measurements established and existing in the organization at the same time minimizing the cost for data reconfiguration. However, in the course of our research new ways of data collection were introduced, which improved the practice and allowed for more accurate predictions.

Based on our experience and evaluation of these methods at Ericsson we can recommend using these methods and adjusting them to local needs for particular organizations. The methods are intended to support projects which are structured around work packages and not sub-projects. Our further work is focused on monitoring the performance of these methods in a larger number of projects and further evaluating their robustness. We also intend to deploy these methods to other departments and organizations to check their external validity.

Acknowledgements. We would like to thank Ericsson AB for support in the study, in particular the experts we have the possibility to work with. We would like to thank the Software Architecture Quality Center for support in this study.

References

[1] W. W. Agresti and W. M. Evanco. Projecting software defects from analyzing ada designs. *Software Engineering, IEEE Transactions on*, 18(11):988–997, 1992.

[2] S. Amasaki, T. Yoshitomi, O. Mizuno, Y. Takagi, and T. Kikuno. A new challenge for applying time series metrics data to software quality estimation. *Software Quality Journal*, 13:177–193, 2005.

[3] T. Ball and N. Nagappan. Static analysis tools as early indicators of pre-release defect density. In *27th International Conference on Software Engineering*, pages 580–586, St. Louis, MO, USA, IEEE, 2005.

[4] S. Chulani. Constructive quality for defect density prediction: COQUALMO. In *International Symposium on Software Reliability Engineering*, pages 3–5, 1999.

[5] S. D. Conte, H. E. Dunsmore, and V. Y. Shen. *Software Engineering Metrics and Models*. Benjamin-Cummings, Menlo Park CA, 1986.

[6] G. H. Dunteman. *Principal Component Analysis*. SAGE Publications, 1989.

[7] A. L. Goel. Software reliability models: Assumptions, limitations, and applicability.

[8] D. Houston, D. Buettner, and M. Hecht. Dynamic COQUALMO: Defect profiling over development cycles. In *ICSP*, pages 161–172, 2009.

[9] M. Klaes, H. Nakao, F. Elberzhager, and J. Munch. Support planning and controlling of early quality assurance by combining expert judgement and defect data – a case study. *Empirical Software Engineering*, 2009.

[10] L. M. Laird and M. C. Brennan. *Software measurement and estimation: a practical approach*. John Wiley and Sons, Hoboken, N.J., 2006.

[11] P. L. Li, J. Herbsleb, and M. Shaw. Forecasting field defect rates using a combined time-based and metrics-based approach: a case study of OpenBSD. In *The 16th IEEE International Symposium on Software Reliability Engineering*, page 10. IEEE, 2005.

[12] Y. K. Malaiya and J. Denton. Module size distribution and defect density. In *11th International Symposium on Software Reliability Engineering*, pages 62–71, San Jose, CA, USA, 2000.

[13] K. Matsumoto, K. Inoue, T. Kikuno, and K. Torii. Experimental evaluation of software reliability growth models. In *The 18th International Symposium on Fault-Tolerant Computing, 1988*, pages 148–153, Tokyo, Japan, IEEE, 1988.

[14] P. Mohagheghi, R. Conradi, O. M. Killi, and H. Schwarz. An empirical study of software reuse vs. defect-density and stability. In *26th International Conference on Software Engineering*, pages 282–291. IEEE, 2004.

[15] M. Neil and N. Fenton. Predicting software quality using bayesian belief networks. In *21st Annual Software Engineering Workshop*, pages 217–230, NASA Goddard Space Flight Centre, 1996.

[16] A. M. Neufelder. How to predict software defect density during proposal phase. In *National Aerospace and Electronics Conference*, pages 71–76, Dayton, OH, USA, 2000.

[17] T. J. Ostrand, E. J. Weyuker, and R. M. Bell. Predicting the location and number of faults in large software systems. *IEEE Transactions on Software Engineering*, 31(4):340–355, 2005.

[18] H. Petersson, T. Thelin, P. Runeson, and C. Wohlin. Capture-recapture in software inspections after 10 years research: Theory, evaluation and application. *Journal of Systems and Software*, 72:249–264.

[19] M. Staron and W. Meding. Predicting weekly defect inflow in large software projects based on project planning and test status. *Information and Software Technology*, 50(7–8):782–796, 2009.

[20] M. Staron, W. Meding, and C. Nilsson. A framework for developing measurement systems and its industrial evaluation. *Information and Software Technology*, 51(4):721–737, 2009.

[21] P. Tomaszewski and L. Lundberg. Software development productivity on a new platform: an industrial case study. *Information and Software Technology*, 47(4):257–269, 2005.

[22] P. Tomaszewski and L. Lundberg. The increase of productivity over time: an industrial case study. *Information and Software Technology*, 48(9):915–927, 2006.

[23] R. E. Walpole. *Probability and statistics for engineers and scientists*. Prentice Hall, Upper Saddle River, NJ, 7th edition, 2002.

[24] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in Software Engineering: An Introduction*. Kluwer Academic Publishing, 2000.