Сеньков А.В. Технический, организационный и инфраструктурный долг в разработке программного обеспечения и автоматизированных систем // Международный журнал информационных технологий и энергоэффективности. – 2020. – Т. 5 № 2(16) с. 36–42

Международный журнал информационных технологий и энергоэффективности

Сайт журнала:
http://www.openaccessscience.ru/index.php/ijcse/

УДК 004

# ТЕХНИЧЕСКИЙ, ОРГАНИЗАЦИОННЫЙ И ИНФРАСТРУКТУРНЫЙ ДОЛГ В РАЗРАБОТКЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ И АВТОМАТИЗИРОВАННЫХ СИСТЕМ

**Сеньков А.В.**
*Филиал ФГБОУ ВО «Национальный исследовательский университет «МЭИ» в г. Смоленске, Россия, (214013, г. Смоленск, Энергетический проезд, 1), e-mail: a.v.senkov@mail.ru.*

**Архитекторы и разработчики программного обеспечения и автоматизированных систем часто употребляют термин «технический» долг как некоторые отложенные задачи, которые могут привести к негативным последствиям в будущем. В настоящей статье предпринята попытка разобрать сущность долга, а также выполнить классификацию видов долга. Также проведена аналогия между управлением долгом и управлением рисками.**

Ключевые слова: архитектура, программное обеспечение, автоматизированные системы, технический долг, организационный долг, риски.

# TECHNICAL, ORGANIZATIONAL, AND INFRASTRUCTURAL DEBT IN A DEVELOPMENT OF SOFTWARE AND AUTOMATED SYSTEMS

**Senkov A.V.**
*Smolensk Branch of the National Research University "Moscow Power Engineering Institute", Smolensk, Russia (214013, Smolensk, Energetichesky proezd, 1), e-mail: a.v.senkov@mail.ru.*

**Architects and developers of software and automated systems often use the term "technical" debt as some deferred tasks that could lead to negative consequences in the future. This article attempts to analyze the nature of debt, as well as classify types of debt. An analogy is also drawn between debt management and risk management.**

Keywords: architecture, software, automated systems, technical debt, organizational debt, risks.

В 1992 году У.Каннингэм (Ward Cunningham) в своем докладе The WeCash Portfolio Management System на конференции OOPSLA (Conference on Object-Oriented Programming Systems, Languages, and Applications) ввёл понятие технического долга [1]. Следует отметить что именно в рамках доклада не выделяется именно технический долг. Вводится метафора долга в целом, который, однако, интерпретируется как незрелость программного кода. Поставка незрелого программного кода, или программного кода «с долгом» само по себе не приводит к негативным последствиям. Опасность возникает только в том случае, если долг не погашен. Он имеет свойство накапливаться, что в конечном итоге может привести к значительным трудозатратам, которые команда будет тратить на поддержку программного обеспечения.

В результате ряда выступлений и публикации ряда работ, например [2, 3], такой долг стал называться техническим поскольку действительно, причина его возникновения лежит в технической сфере, в коде программного обеспечения.

Что кардинально изменилось с 1992 года, когда такой долг был определен. В конце 80-х – начале 90-х годов получили значительное развитие и распространение многие методологии разработки программного обеспечения. В первую очередь, это гибкие методологии. Основная ценность в соответствии с Agile Manifest [4] в таких методология начинает переноситься на работающий продукт. Кроме того, значительное распространение получают прототипы, называемые MVP (minimum viewable product), которые, зачастую, разрабатываются малыми командами, после чего выводятся в продуктивную эксплуатацию.

Безусловно, в большинстве программных продуктов, прошедших путь через MVP, будет зафиксирован технический долг, однако, одновременно с продуктом, чаще всего, развивается и команда. Состав команды меняется, она пополняется новыми ролями, возникает потребность в новых функциях, изменении используемых организационных инструментов, подходов, практик и т.д.

При этом откладывание таких изменений в долгий ящик может также значительно усложнить работу по разработке и дальнейшей поддержке программного обеспечения. Каким образом можно поименовать такого рода недоработки?

Во-первых, эти недоработки заметны и могут быть идентифицированы, особенно подготовленным специалистом – специалистом с большим опытом работы в аналогичных командах / на аналогичных проектах.

Во-вторых, действия по устранению указанных недостатков действительно могут быть отложены, и, вероятно, это не приведет к значительным последствиям в краткосрочной перспективе.

В-третьих, сохранение этих недостатков принесет возможные потери в долгосрочной перспективе.

По аналогии с финансовыми долгами, предложенной изначально в [1], указанные недоработки в соответствии с их перечисленными особенностями, также можно отнести к одному из видов долга, однако, поскольку этот «долг» не относится непосредственно к программному коду, называть его техническим нецелесообразно. Справедливости ради, следует отметить, что в некоторых источниках, например [5, 6], такого вида задолженность относится к разряду технических, хотя непосредственно к программному коду отношения не имеет.

Как правило, автоматизированные системы рассматривают в виде триплета [7], включающего программное обеспечение, аппаратное обеспечение и организационное обеспечение. Если рассматривать долг сквозь призму видов обеспечения автоматизированных систем (частей автоматизированных систем), то и долг может быть разделен исходя из принадлежности к одной из таких частей. Исходя из этого, целесообразно сформировать следующую классификацию долга.

1. Технический долг. Поскольку термин «технический долг» можно считать прочно вошедшим в обиход специалистов по программной инженерии, целесообразно этим термином обозначать любой долг, относящийся к программному обеспечению, хотя более точным определением, вероятно, стало бы определение «Программный долг».

2. Инфраструктурный или аппаратный долг. Ранее явно не выделявшийся вид долга, к которому можно в зависимости от конкретного проекта или команды относить или долг по улучшению аппаратного обеспечения, на котором развертывается или используется автоматизированная система. К такого рода долгу можно отнести недостаточную обеспеченность серверными мощностями / средствами хранения информации и т.д. Например, недостаточное резервирование серверных мощностей значительно повышает вероятность отказа системы с сопутствующими невосстановимыми потерями.
3. Организационный долг – совокупность допущений и ограничений, принятых в организации и/или команде, которые в результате развития организации/команды могут значительным образом повлиять на стабильность организации/команды или разрабатываемой ими системы. Организационный долг является наиболее сложным для идентификации не только по причине того, что для идентификации нужен специалист с широким кругозором в смысле организации труда, но и ввиду отсутствия точных рецептов организации труда в такой отрасли как разработка автоматизированных систем.

Помимо рассмотренной классификации по частям автоматизированной системы, долг может также быть классифицирован по ролям, накапливающим этот долг. Заранее определимся что будем рассматривать только роли, относящиеся непосредственно к разработке автоматизированных систем. Кроме того, возьмем расширенный перечень ролей, характерных для различных методологий разработки автоматизированных систем [8, 9, 10], характерных для большинства организаций и команд.

Рассмотрим возможность указанных ролей накапливать различные виды долга (таблица 1).

Таблица 1 – Роли и их возможности по накоплению различных видов долга

| № п/п | Роль | Технический долг | Инфра-структур-ный долг | Организа-ционный долг |
|---|---|---|---|---|
| **1** | **2** | **3** | **4** | **5** |
| 1 | Директор, директор по информационным технологиям (CIO) – руководитель | - | + | ++ |
| 2 | Технический директор (CTO) | ++ | ++ | + |
| 3 | Директор по инфраструктуре | - | ++ | ++ |
| 4 | Директор по информационной безопасности | + | + | ++ |
| 5 | Операционный директор (COO, Scrum Master) | - | - | ++ |
| 6 | Руководитель продукта (Product Owner) | + | + | + |
| 7 | Руководитель программы (Program Manager, Product Development Manager, System Engineering Manager и др.) | + | + | ++ |

| № п/п | Роль | Технический долг | Инфра-структур-ный долг | Организа-ционный долг |
|---|---|---|---|---|
| **1** | **2** | **3** | **4** | **5** |
| 8 | Руководитель разработки (Team Lead) | ++ | + | + |
| 9 | Архитектор (Enterprise Architect, Solution Architect и др.) | ++ | ++ | - |
| 10 | Аналитик (бизнес, системный, не продуктовый) | + | - | - |
| 11 | Разработчик | ++ | - | - |
| 12 | Специалист по качеству (Тестировщик) | + | - | - |
| 13 | Системный администратор | - | + | - |
| 14 | UX | + | - | + |
| 15 | Технический писатель | + | - | + |

В таблице:

«-» - означает что для роли не характерно накопление соответствующего вида долга;

«+» - означает что роль ограниченно может накапливать соответствующий долг

«++» - означает что роль в значительной степени может накапливать соответствующий долг.

Таким образом, различные роли могут по-разному влиять на накопление долга различного вида.

Кроме того, при рассмотрении причин накопления долга, можно отметить что организации/команды на различных этапах своего развития могут накапливать долг разного рода в разной степени. Этот вопрос, очевидно, касается траекторий развития организаций/команд и должен быть проработан отдельно.

Однако, каким образом можно пробовать управлять различного рода долгом. Если долг может нести для организации или команды потери в определенной перспективе, то можно провести аналогию между долгом и рисками. При этом, для рисков разработано достаточно большое количество методов, моделей и средств для управления ими [11, 12, 13, 14]. Если рассматривать управление долгом с точки зрения процесса управления рисками, представленного в [13], многие задачи управления долгом становятся очевидными (см. рисунок 1). Однако, каждый из этапов процесса управления долгом должен быть рассмотрен и детализирован отдельно.

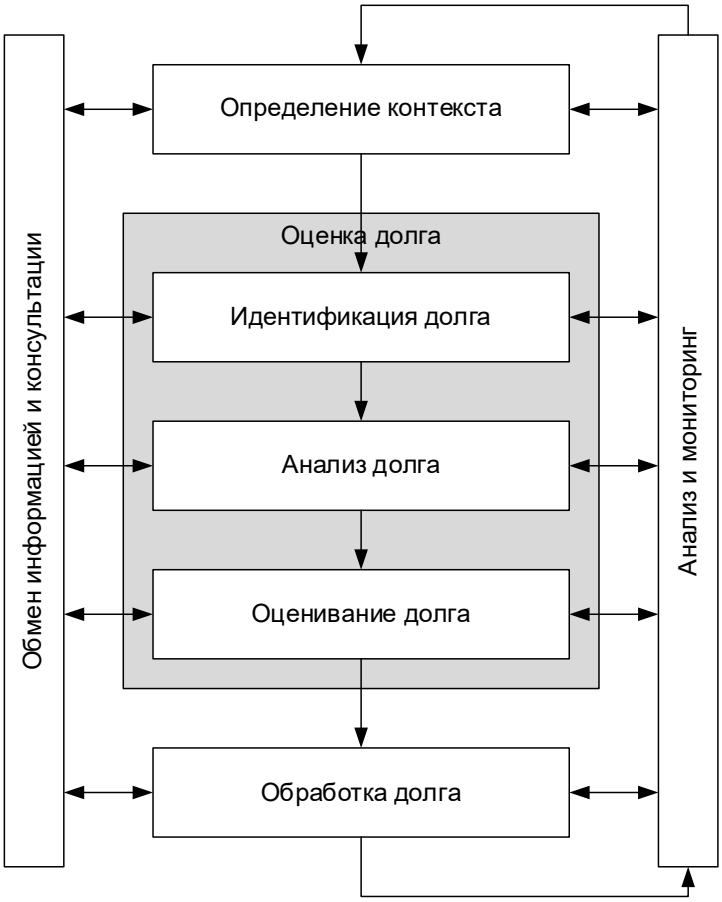


Рисунок 1 – Обобщенный процесс управления долгом

Таким образом, долг, накапливаемый в рамках процесса разработки автоматизированных систем можно разделить на технический (программный), инфраструктурный (аппаратный) и организационный. Различные роли, участвующие в разработке автоматизированных систем, в силу своих обязанностей могут допускать накопление различного рода долга в разной степени. Управление долгом можно рассматривать по аналогии с управлением рисками.

**Список литературы**

1. Cunningham, W.: The WyCash Portfolio Management System. In: Addendum to the Proceedings on Object-Oriented Programming Systems, Languages, and Applications, OOPSLA 1992, pp. 29–30. ACM, New York (1992);
2. Fowler M. TechnicalDebt [сайт]. URL: https://www.martinfowler.com/bliki/TechnicalDebt.html (дата обращения: 11.06.2020);
3. Yli-Huumo J., Maglyas A., Smolander K. (2014) The Sources and Approaches to Management of Technical Debt: A Case Study of Two Product Lines in a Middle-Size Finnish Software Company. In: Jedlitschka A., Kuvaja P., Kuhrmann M., Männistö T., Münch J., Raatikainen M. (eds) Product-Focused Software Process Improvement. PROFES 2014. Lecture Notes in Computer Science, vol 8892. Springer, Cham

4. Galbraith K. Two Kinds of Tech Debt and How to Pay It Down [сайт] URL: https://blog.kylegalbraith.com/2018/10/22/two-kinds-of-tech-debt-and-how-to-pay-it-down/ (дата обращения: 11.06.2020);
5. Agile-манифест разработки программного обеспечения [сайт] URL: https://agilemanifesto.org/iso/ru/manifesto.html (дата обращения: 11.06.2020);
6. Yli-Huumo J., Maglyas A., Smolander K. (2016) The Effects of Software Process Evolution to Technical Debt—Perceptions from Three Large Software Projects. In: Kuhrmann M., Münch J., Richardson I., Rausch A., Zhang H. (eds) Managing Software Process Evolution. Springer, Cham DOI: 10.1007/978-3-319-31545-4_15
7. Информационные технологии : учеб. пособие / Г.Н. Исаев. – М. : Издательство «Омега-Л», 2012.
8. Руководство к Своду знании по управлению проектами (Руководство PMBOK®) -- Шестое издание - PMI, 2017.
9. A. Stellman, J. Greene Learning Agile. – O'Reilly Media, Inc., 2014
10. J. Sutherland, J.O. Coplien A Scrum Book. – Pragmatic Bookshelf., 2019
11. Королев В.Ю., Беннинг В.Е., Шоргин С.Я. Математические основы теории риска: Учебн. пособ. – 2–е изд., перераб. и доп. – М.: Физматлит, 2011.
12. ISO 31000:2009 – Risk management Principles and guidelines
13. Борисов В.В., Сеньков А.В. Интеллектуальное управление рисками в сложных организационно-технических системах // Информационные технологии, 2011, № 10. – С. 47–51.
14. Сеньков А.В. Управление рисками: интеллектуальные модели, методы, средства. – Смоленск: Универсум, 2016

**References**

1. Cunningham, W.: WyCash Portfolio Management System. In the book: Supplement to the materials "Object-oriented programming systems, languages and applications", OOPSLA 1992, p. 29-30. AFM, New York (1992);
2. Fowler M. TechnicalDebt [site]. URL: https://www.martinfowler.com/bliki/TechnicalDebt.html (accessed date: 06/11/2020);
3. Yli-Huumo J., Maglyas A., Smolander K. (2014) Sources and approaches to technical debt management: a case study of two product lines in a medium-sized Finnish software company. In: Jedlitschka A., Kuvaja P., Kuhrmann M., Männistö T., Münch J., Raatikainen M. (eds) Improving the product-oriented software development process. PROFES 2014. Lecture Notes in Computer Science, Volume 8892. Springer, Cham
4. Galbraith K. Two types of technical debt and how to pay it off [site] URL: https://blog.kylegalbraith.com/2018/10/22/two-kinds-of-tech-debt-and-how-pay- down / (circulation date: 06/11/2020);
5. Agile-manifest of software development [site] URL: https://agilemanifesto.org/iso/ru/manifesto.html (accessed: 06/11/2020);
6. Juli-Uumo J., Maglyas A., Smolander K. (2016) The influence of the evolution of software processes on technical debt - the perception of three major software projects. In the book: Kurmann M., Münch J., Richardson I., Rausch A., Zhang H. (eds.) Software Development Management. Springer, Cham DOI: 10.1007 / 978-3-319-31545-4_15

7. Information technology: textbook. allowance / G.N. Isaev. - M.: Publishing house "Omega-L", 2012.
8. Guide to the Project Management Body of Knowledge (PMBOK® Guide) - Sixth Edition - PMI, 2017.
9. A. Stellman, J. Greene Learning Agile. – O'Reilly Media, Inc., 2014
10. J. Sutherland, J.O. Coplien A Scrum Book. – Pragmatic Bookshelf., 2019
11. Korolev V.Yu., Benning V.E., Shorgin S.Ya. Mathematical foundations of risk theory: Textbook. benefits - 2nd ed., Rev. and add. - M.: Fizmatlit, 2011.
12. ISO 31000: 2009 - Principles and guidelines for risk management
13. Borisov V.V., Senkov A.V. Intelligent risk management in complex organizational and technical systems // Information Technologies, 2011, No. 10. - P. 47-51.
14. Senkov A.V. Risk management: intellectual models, methods, tools. - Smolensk: Universum, 2016.