توسعه هستی شناسی و تجزیه و تحلیل مدل های چرخه دوام توسعه نرم افزاری

چکیده

در حوزه فعلی توسعه نرم افزاری، تعداد زیادی از مدل های چرخه دوام برای توسعه نظام مند نرم افزار و پروژه های رایانه ای، نظیر مدل آبشار، مدل آبشار تکرارشوند، مدل نمونه، مدل مارپیچی و غیره در دسترس هستند. این مدل ها دارای ویژگی های خاص خود بوده و برای موقعیت خاصی از توسعه نرم افزاری و نرم افزار مناسب هستند. ممکن است مشخص شود که یک مدل چرخه دوام نرم افزار بسته به محیط توسعه پربازده تر از دیگری باشد. در این مقاله، تلاش شده است تا از این جنبه مدل های مختلف چرخه دوام نرم افزاری مورد تجزیه و تحلیل قرار گیرند. علاوه بر این، تلاش هایی نیز برای توسعه هستی شناسی دسته های مختلفی از پروژه های نرم افزاری و بررسی مدل های چرخه دوام نرم افزاری بر اساس این دسته بندی ها صورت گرفته است.

کلمات کلیدی: دسته ها، هستی شناسی، طبقه بندی، پروژه نرم افزاری، SDLC

1. مقدمه

پروژه نرم افزاری، بدون توجه به این که کوچک یا بزرگ باشند، ازمیان مراحل مشخصی عبور می کنند که با یکدیگر به عنوان چرخه دوام توسعه نرم افزاری (SDLC) شناخته می شوند. پنج مرحله وجود دارد که بخشی از SDLC به شمار می روند. این مراحل عبارت هستند از: تعریف ملزومات، طراحی، کد گذاری، آزمودن، و نگهداری.

مدل های SDLC بر اساس مراحل مختلف SDLC، ترتیبی که آنها طبق آن روی می دهند و تعامل بین آنها ایجاد می شوند. خروجی ناشی از هر مرحله به عنوان ورودی مرحله بعد عمل می کند. در بخش بعد، به بحث در مورد برخی از مدل های SDLC خواهیم پرداخت. مکن است ثابت شود که یکی از مدل های چرخه دوام توسعه نرم افزاری بسته به محیط توسعه، پربازده تر از مدل های دیگر باشد. در این مقاله، تلاش شده است تا مدل های مختلف SDLC از این دیدگاه مورد تجزیه و تحلیل قرار گیرند. دسته بندی هستی شناسی پروژه های نرم افزاری مختلف نیز در این مقاله ارائه می شود.

به غیر از مقدمه ارائه شده در بخش 1، این مقاله به صورت زیر سازماندهی شده است. در بخش 2، خلاصه ای در مورد مدل های مختلف SDLC ارائه شده است. تحلیل تناسب مدل های گوناگون در بخش 3 انجام شده است. در بخش 4، دسته بندی کلی انواع مختلف پروژه های نرم افزاری ارائه شده است. در این بخش، به هستی شناسی نیز پرداخته می شود. بخش 5 به ارائه مشاهدات ما از کار پرداخته و در بخش 6 نیز جمع بندی صورت می گیرد.

2. مدل های SDLC

A. مدل آبشار

مدل آبشار رویکردی کلاسیک است که در مهندسی نرم افزار برای اطمینان از موفقیت پروژه به صورت گسترده ای مورد استفاده قرار می گیرد.

B. مدل نمونه

مدل نمونه به سرعت نمونه کاری را توسعه می دهد که به لحاظ کارکردی با یکی از مؤلفه های پروژه برابر می باشد.

RAD .C

RAD مفهومی است که بر طبق آن محصولات می توانند با استفاده از تکنیک های خاصی با کیفیت بالاتری سریع تر توسعه یابند.

D. مدل فزاینده

این مدل اجرای ناقصی از کل سیستم را ارائه می دهد. سپس، به تدریج بر کارکرد آن اضافه می کند.

E. مدل مارپیچی

این مدل ویژگی های مدل نمونه و مدل آبشار را با یکدیگر ترکیب می کند؛ و سپس، تحلیل خطر، RAD 4gl و نمونه ای از مدل آبشار را اضافه می نماید.

F. مدل XP (برنامه ریزی نهایت)

مدل XP برای تیم های کوچک تا متوسط استفاده می شود که با ملزومات نامشخص یا به سرعت متغیر به توسعه نرم افزار می پردازند. کدگذاری مهم ترین کار در یک پروژه نرم افزاری می باشد.

G. مدل ازدحامی

این مدل شبیه به سایر مدل های چرخه دوام است که از توسعه تکرارشونده برای رسیدگی به ملزومات متغیر استفاده می کند، اما در آن، تکرارها معمولا سی روز طول می کشند.

مدل های SDLC مختلف دارای خصوصیات و الزامات منحصر به فرد خود هستند. بر اساس این جنبه ها، این مقاله به ارائه تحقیقی تحلیلی در مورد تناسب مدل های مختلف SDLC برای استفاده در توسعه پروژه های نرم افزاری می پردازد. علاوه بر این، برای دسته های مختلف پروژه های نرم افزاری هستی شناسی صورت می گیرد و بر اساس این

دسته بندی ها، مدل های چرخه دوام نرم افزاری مورد بررسی قرار می گیرند. این هستی شناسی در زبان توسعه هستی شناسی شبکه معنایی[1]، زبان هستی شناسی شبکه[2] ارائه داده شده است.

جدول 1 به بحث در مورد تناسب مدل های مختلف SDLC برای استفاده در توسعه پروژه های نرم افزاری می پردازد.

جدول 1. تحلیل تناسب مدل های مختلفSDLC

SDLC	نوع موقعیتی که برای آن مناسب است
آبشار	-ملزومات ثابت -کار به شیوه ای خطی به جلو پیش می رود. -بدون محدودیت زمانی -می تواند برای پروژه های با خطر بالا نیز استفاده شود.
XP	-پروژه های وچک -نماینده تمام وقت کاربر. -اگر تیم هزینه ها و زمان بندی را پیش بینی نماید، کار نمی کند.
ازدحام	-پروژه های توسعه گسترده. -پروژه هایی که از تکنولوژی شی گرا استفاده می کنند.
RAD	-محدودیت زمانی وجود دارد. -خطر زیاد بالا نیست. -پروژه های کوچک تا متوسط.
نمونه	-توسعه دهندگان چیزی را به سرعت ایجاد می کنند. -نمی تواند درصد بالایی از خطرات را مدیریت نماید.
فزاینده	-خطرات تکنیکی بالا. -محدودیت زمانی وجود دارد.
مارپیچی	-سیستم و نرم افزار با مقیاس بزرگ. -سطح اعتبار و اطمینان بالایی دارد. -درک و واکنش نسبت به خطرات در سطحی تکاملی صورت می گیرد.

[1] Semantic Web ontology development language

[2] Web Ontology Language (OWL)

3. دسته بندی کلی پروژه های نرم افزاری و هستی شناسی آنها

A. دسته بندی پروژه های نرم افزاری

پروژه های نرم افزاری بر اساس ویژگی های خود مثل کاربرد پروژه نرم افزاری، تعداد خطوط کدها، تعداد مؤلفه های مورد استفاده، شرایط جغرافیایی، بخش سخت افزاری، نیازمندی های و کاربر و غیره در دسته های مختلف دسته بندی می شوند.

مبانی طبقه بندی پروژه های نرم افزاری در تصویر 1 نشان داده شده اند. بر اساس این مبنا، توضیح دسته های مختلف به زیر ارائه می شود.

1. دسته های سطح اول

i. پروژه نرم افزار کاربردی از برنامه های مستقل تشکیل می شود که نیاز تجاری خاصی را برطرف می نماید.

ii. پروژه نرم افزار سیستم مجموعه ای از برنامه های نوشته شده برای خدمات رسانی به سایر برنامه ها می باشد.

iii. پروژه نرم افزاری تثبیت شده در سیستم یا محصولی مستقر شده و برای اجرا و کنترل ویژگی ها و کارکردها برای کاربر و خود سیستم مورد استفاده قرار می گیرد.

B. بازنمایی هستی شناختی طبقه بندی پیشنهادی

بازنمایی هستی شناختی به معنای بازنمایی سلسله مراتبی مبانی طبقه بندی به صورت نمودار می باشد.

بر اساس مبانی طبقه بندی بالا در مورد دسته های نرم افزاری، این مقاله با استفاده از کتابخانه JENA در JAVA به ارائه هستی شناسی می پردازد. هستی شناسی در زبان OWL طراحی شده و سپس با استفاده از نرم افزار Altova به صورت نمودار نشان داده می شود. هستی شناسی طراحی شده به خوبی شکل می گیرد.

تصویر 2 بازنمایی هستی شناختی اولین سطح از مبانی طبقه بندی ارائه شده را نشان می دهد.

سه دسته اصلی پروژه های نرم افزاری برنامه کاربردی، سیستم و تثبیت شده هستند. همانطور که در تصویر نشان داده شده است، پروژه نرم افزاری در رأس قرار گرفته و برنامه کاربردی، سیستم و تثبیت شده جز زیرشاخه های آن هستند.

تصویر 3 بازنمایی هستی شناسی سطح دوم و یک دسته از سطح سوم مبانی طبقه بندی ارائه شده را نشان می دهد که در آن، پروژه نرم افزاری دسته اصلی، سیستم و زیرمجموعه آن "پیچیدگی" و پس از آن، زیر مجموعه هایش "کمتر، بیشتر و خیلی زیاد" هستند.

4. مشاهده

پس از تجزیه و تحلیل مدل های SDLC و دسته بندی نرم افزار، دریافتیم که مدل مناسبی برای انواع مختلف نرم افزار وجود دارد. جدول مشاهده مدل های SDLC با توجه به مبانی طبقه بندی ارائه شده در جدول 2 نشان داده شده است.

جدول 2. جدول مشاهده مدل های SDLC

شماره	موقعیت ها	مدل های SDLC
1.	در جایی که نوع نرم افزار کاربردی، اندازه متوسط، خطر کم، پیچیدگی کمتر، کاربر یک نفر، استاندارد آن استاندارد، بخش سخت افزاری PC، قابلیت اطمینان پایین و زمان غیرفشرده باشد.	آبشار/ XP
2.	در جایی که نوع نرم افزار کاربردی، اندازه بزرگ، خطر کم، پیچیدگی کمتر، کاربر یک نفر، استاندارد آن استاندارد، بخش سخت افزاری PC، قابلیت اطمینان پایین و زمان غیرفشرده باشد.	آبشار/ ازدحامی
3.	در جایی که نوع نرم افزار سیستم، اندازه کوچک، خطر کم، پیچیدگی کمتر، کاربر یک نفر، استاندارد آن استاندارد، بخش سخت افزاری PC، قابلیت اطمینان بالا و زمان فشرده باشد.	RAD/ نمونه
4.	در جایی که نوع نرم افزار سیستم، اندازه کوچک تر، خطر بالا، پیچیدگی زیاد،	فزاینده/ مارپیچی

	کاربر یک نفر، استاندارد آن استاندارد، بخش سخت افزاری PC، قابلیت اطمینان بالا و زمان فشرده باشد.	
5.	در جایی که نوع نرم افزار تثبیت شده، اندازه بزرگ، خطر کم، پیچیدگی بیشتر، کاربر یک نفر، استاندارد آن استاندارد، بخش سخت افزاری PC، قابلیت اطمینان بالا و زمان فشرده باشد.	ازدحام/ RAD
6.	در جایی که نوع نرم افزار تثبیت شده، اندازه بزرگ، خطر زیاد، پیچیدگی کمتر، کاربر یک نفر، استاندارد آن استاندارد، بخش سخت افزاری پردازنده مرکزی، قابلیت اطمینان بالا و زمان غیرفشرده باشد.	مارپیچی/ فزاینده
7.	در جایی که نوع نرم افزار کاربردی، اندازه کوچک، خطر زیاد، پیچیدگی کمتر، کاربر یک نفر، استاندارد آن استاندارد، بخش سخت افزاری ترمینال، قابلیت اطمینان متوسط و زمان فشرده باشد.	فزاینده/ مارپیچی
8.	در جایی که نوع نرم افزار کاربردی، اندازه بزرگ، خطر زیاد، پیچیدگی کمتر، کاربر شرکت، استاندارد آن استاندارد، بخش سخت افزاری PC، قابلیت اطمینان بالا و زمان فشرده باشد.	مارپیچی/ فزاینده
9.	در جایی که نوع نرم افزار کاربردی، اندازه بزرگ، خطر کم، پیچیدگی زیاد، کاربر شرکت، استاندارد آن استاندارد، بخش سخت افزاری PC، قابلیت اطمینان پایین و زمان فشرده باشد.	ازدحام/ RAD
10.	در جایی که نوع نرم افزار کاربردی، اندازه کوچک، خطر متوسط، پیچیدگی بیشر، کاربر یک نفر، استاندارد آن استاندارد، بخش سخت افزاری ترمینال، قابلیت اطمینان متوسط و زمان فشرده باشد.	RAD/ نمونه

5. جمع بندی

در این مقاله، تلاش شده است تا مدل های مختلف SDLC با توجه به تناسب آنها برای توسعه انواع مختلف پروژه های نرم افزاری بسته به محیط توسعه مورد تجزیه و تحلیل قرار گیرند. همچنین، تلاش شده است تا دسته بندی

گسترده ای صورت گرفته و بازنمایی هستی شناختی انواع مختلف پروژه های نرم افزاری آن صورت گیرد. مشاهداتی نیز صورت گرفته است تا مدل های SDLC مناسب برای محیط های توسعه متنوع تعیین شود.

REFERENCES

[1] Abrahamsson, Pekka, Salo, Outi, Ronkainen, Jussi & Warsta, Juhani, "Agil software development methods. Review and analysis", Espoo, VTT Publications, 2002, pp. 107-110.

[2] Executive Brief, "Which Life Cycle Is Best For Your Project?", www.executivebrief.com/project.../best-project-life-cycle/ - United States, accessed on November 14, 2010.

[3] H. Gumuşkaya, "Core Issues Affecting Software Architecture in Enterprise Projects", World Academy of Science, Engineering and Technology 9- 2005. pp 32-37.

[4] I. Sommerville, "Software Engineering, Software Engineering", 7th edition, Addison-Wesley,2004, Reading, MA.

[5] J. Rothman, "What Lifecycle? Selecting the Right Model for Your Project", Cutter IT Journal, Vol 21, #5, May 2008.

[6] K. Schwalbe, "Information Technology Project Management", 6th edition, Cengage Learning, 2009.

[7] M. A. A. Ahmar, "Rule Based Expert System For Selecting Software Development Methodology", Journal of Theoretical and Applied Information Technology, 2005, pp 143-148.

[8] M. Cusumano, A. MacCormack, C. F. Kemerer, W. Crandall, "A Global Survey of Software Development Practices", MIT, Harvard University, University of Pittsburgh, Hewlett-Packard, Version 3.1 June 17, 2003, Forthcoming, IEEE Software.

[9] PMP, Fellow PMI, R. D. Archibald, and APM/IPMA, "Life Cycle Models For High-Technology Projects–Applying Systems Thinking To Managing Projects", Presented to the PMI-Central Iowa Chapter, Professional Development Day, Polk County Convention Complex, Des Moines, Iowa, October 17, 2003.

[10] R. D. Archibald and V. I. Voropaev, "Commonalities and Differences in Project Management Around The World: A Survey Of Project Categories And Life Cycles", 17th IPMA World Congress, Moscow, June 4-6 2003.

[11] R. D. Banker, S. M. Datar, and C. F. Kemerer, "Factors Affecting Software Maintenance Productivity: An Exploratory Study", Carnegie Mellon University and Sloan School of Management Massachusetts Institute of Technology. Management Science, Vol 37, No. 1, printed in USA, January 1991, Pp 160-175.

[12] R. S. Pressman, "Software Engineering, A Practitioner's Approach", Sixth Edition, Mc Graw. Hill, 2005.

[13] R Lagerstr, L. M. V. W¨urtemberg, H. Holm, and O. Luczak, "Identifying Factors Affecting Software Development Cost", Industrial Information and Control Systems, the Royal Institute of Technology Stockholm, Sweden, Apr. 2010.

[14] S. Bhattacharjee, "Software Development Software Development Life Cycle", MOF, College of Agricultural Banking, RBI, PUNE, ab.org.in/Lists/ Knowledge%20Bank/ Attachments/ 83/, accessed on November 12, 2010.