دانشگاه پیام‌نور

دانشکده فنی مهندسی

پایان‌نامه

برای دریافت کارشناسی‌ارشد

رشته مهندسی کامپیوتر- گرایش نرم‌افزار

گروه مهندسی کامپیوتر و فناوری‌اطلاعات

ارائه‌ی راهکاری مبتنی بر معماری به‌منظور بازپیکربندی پویا و ایمن سیستم‌های نرم‌افزاری مبتنی بر مولفه

محمد غفاری

استاد راهنما:

دکتر حسن حقیقی

استاد مشاور:

دکتر احمد فراهی

تیرماه ۱۳۹۱

به‌نام آن‌که جان را فکرت آموخت

دانشگاه پیام‌نور

دانشکده فنی مهندسی

مرکز شمیرانات

پایان‌نامه

برای دریافت کارشناسی‌ارشد

رشته مهندسی کامپیوتر- گرایش نرم‌افزار

گروه مهندسی کامپیوتر و فناوری اطلاعات

ارائه‌ی راهکاری مبتنی بر معماری به‌منظور بازپیکربندی پویا و ایمن سیستم‌های نرم‌افزاری مبتنی بر مولفه

محمد غفاری

استاد راهنما:

دکتر حسن حقیقی

استاد مشاور:

دکتر احمد فراهی

تیرماه ۱۳۹۱

تاریخ .....
شماره .....
پیوست .....

جمهوری اسلامی ایران
وزارت علوم تحقیقات و فناوری

دانشگاه پیام نور

دانشگاه پیام نور استان تهران

مرکز شمیرانات

تصویب نامه

پایان نامه کارشناسی ارشد رشته مهندسی کامپیوتر(نرم افزار)

تحت عنوان:

"ارائه راهکاری مبتنی بر معماری به منظور بازپیکربندی پویا و ایمن سیستم های نرم افزاری مبتنی بر مولفه"

تاریخ دفاع :۱۳۹۱/۰٤/۳۱ ساعت: ۱٦-۱۵

نمره:.....۱۹/۵..... درجه ارزشیابی: ..عالی.....

هیات داوران:

داوران	نام و نام خانوادگی	مرتبه علمی	امضاء
استاد راهنما	دکتر حسن حقیقی	استادیار	
استاد مشاور	دکتر احمد فراهی	استادیار	
استاد داور	دکتر اکبر فرهودی نژاد	استادیار	
نماینده گروه	دکتر محمد هادی معظم	استادیار	

تهران- بزرگراه ارتش-انتهای بلوار شهید مژدی (اوشان)
خیابان شهید پیروز شفیعی
خیابان یاران-خیابان یاران دوم
دانشگاه پیام نور مرکز شمیرانات
تلفن:٤-۲۲۱۹۵۳۰۳
دورنگار:۲۲٤۸٤۸۳٤
www.shemiranat.tpnu.ac.ir
shemiranat@tpnu.ac.ir

اینجانب محمدغفاری دانشجوی ورودی سال ۱۳۸۹ مقطع کارشناسی‌ارشد رشته مهندسی‌کامپیوتر-نرم‌افزار گواهی می‌نمایم چنانچه در پایان‌نامه خود از فکر، ایده و نوشته دیگری بهره گرفته‌ام با نقل قول مستقیم یا غیر مستقیم منبع و ماخذ آن را نیز در جای مناسب ذکر کرده‌ام. بدیهی است مسئولیت تمامی مطالبی که نقل قول دیگران نباشد بر عهده خویش می‌دانم و جوابگوی آن خواهم بود.

دانشجو تأیید می نماید که مطالب مندرج دراین پایان‌نامه (رساله) نتیجه تحقیقات خودش می‌باشد و در صورت استفاده از نتایج دیگران مرجع آن را ذکر نموده است.

نام و نام‌خانوادگی دانشجو: محمد غفاری
تاریخ و امضا

اینجانب محمد غفاری دانشجوی ورودی سال ۱۳۸۹ مقطع کارشناسی‌ارشد رشته مهندسی‌کامپیوتر-نرم‌افزار گواهی می‌نمایم چنانچه براساس مطالب پایان‌نامه خود اقدام به انتشار مقاله، کتاب، و ... نمایم ضمن مطلع نمودن استاد راهنما، با نظر ایشان نسبت به نشر مقاله، کتاب، و ... و به صورت مشترک و با ذکر نام استاد راهنما مبادرت نمایم.

نام و نام‌خانوادگی دانشجو: محمد غفاری
تاریخ و امضا

کلیه حقوق مادی مترتب از نتایج مطالعات، آزمایشات و نوآوری ناشی از تحقیق موضوع این پایان‌نامه متعلق به دانشگاه پیام‌نور می‌باشد.

تیرماه ۱۳۹۱

سپاسگزارم
از پدر و مادر عزیزم که موفقیت و پیشرفت هر لحظه از زندگی‌ام را، مدیون حمایت‌ها، محبت‌ها و فداکاری‌هایشان هستم.

سپاسگزارم
از استاد گرامی، دکتر حسن حقیقی که راهنمایی و نظرات ارزشمندشان چراغ راه این تحقیق بوده است.

سپاسگزارم
از استاد ارجمند، دکتر احمد فراهی که در بوستان مصفای علم و تربیت و از همان بدو ورود به دانشگاه، با راهنمایی‌ها و صحبت‌های ارزشمندشان پیرامون فعل خواستن، حقیقت را به من هدیه نمودند.

سپاسگزارم
از دیگر عزیزانی که در انجام و تدوین این پایان‌نامه اینجانب را یاری نمودند.

فهرست مطالب

۱- کلیات تحقیق ۱

۱-۱- مقدمه ۲

۱-۲- تعریف مساله ۳

۱-۳- سوالات تحقیق ۶

۱-۴- فرضیه‌ها ۷

۱-۵- پیش‌فرض ها ۷

۱-۶- اهداف تحقیق ۷

۱-۷- کاربردهای متصور از این تحقیق ۸

۱-۸- نوآوری تحقیق ۹

۱-۹- روش انجام تحقیق ۹

۱-۱۰- ساختار پایان‌نامه ۱۲

۲- ادبیات تحقیق ۱۳

۲-۱- مقدمه ۱۴

۲-۲- معماری نرم‌افزار ۱۴

۲-۲-۱- تعریف ۱۵

۲-۲-۲- مفاهیم ابتدایی ۱۸

۲-۲-۲-۱- توسعه‌ی مبتنی بر مولفه ۱۸

۲-۲-۲-۲- مولفه ۱۹
۲-۲-۲-۳- اتصال گر ۱۹
۲-۲-۲-۴- پیکربندی ۲۱
۲-۲-۲-۵- سیستم ۲۲
۲-۲-۲-۶- پورت ۲۲
۲-۲-۲-۷- اتصال ۲۳
۲-۲-۲-۸- ارتباط مرکب ۲۳
۲-۲-۲-۹- سبک معماری ۲۴
۲-۳- تکامل نرم‌افزار ۲۴
۲-۳-۱- خودمدیریتی ۲۵
۲-۳-۲- نگهداری درحال اجرا ۲۷
۲-۳-۳- تفاوت نگهداری نرم‌افزار و تکامل نرم‌افزار ۲۹
۲-۳-۴- تکامل به‌معنای بخشی از فرایند توسعه ۳۰
۲-۴- تکامل پویای نرم‌افزار ۳۱
۲-۴-۱- تغییر محصول نرم‌افزاری ۳۲
۲-۴-۲- تغییرات در حالت اجرای نرم‌افزار ۳۳
۲-۴-۳- تعاریف تکامل پویا ۳۴
۲-۴-۴- انواع تکامل پویا ۳۶
۲-۴-۴-۱- ریز و درشتی تغییرات ۳۷

۲-۴-۴-۲- انواع تغییر ۳۸

۲-۵- مسائل اصلی در تکامل پویا ۴۰

۲-۵-۱- توقف ایمن سیستم در حال اجرا ۴۰

۲-۵-۱-۱- سکون ۴۱

۲-۵-۱-۲- آسودگی ۴۴

۲-۵-۱-۳- راه‌کارهای دیگر در توقف ایمن نرم‌افزار ۴۸

۲-۵-۲- به‌روز رسانی همراه با انتقال وضعیت‌ها ۴۸

۲-۶- جمع‌بندی ۵۱

۳- کارهای مرتبط ۵۲

۳-۱- مقدمه ۵۳

۳-۲- سکون ۵۳

۳-۳- آسودگی ۵۶

۳-۴- روش‌های دیگر ۶۳

۳-۵- ویژگی‌های یک معیار مناسب ۶۵

۳-۶- جمع‌بندی ۶۸

۴- روش پیشنهادی ۶۹

۴-۱- مقدمه ۷۰

۴-۲- مرور صورت‌مسئله ۷۰

۴-۳- نقش معماری ۷۱
۴-۴- تعاریف ۷۳
۴-۴-۱- ارسال غیرمستقیم پیام ۷۳
۴-۴-۲- تغییر معنایی ۷۴
۴-۴-۳- پیکربندی سازگار ۷۵
۴-۴-۴- آرامش ۷۵
۴-۵- فعال‌نمودن آسودگی در تراکنش‌های توزیع‌شده ۷۶
۴-۶- جمع‌بندی ۷۸
۵- ارزیابی روش پیشنهادی ۷۹
۵-۱- مقدمه ۸۰
۵-۲- انتخاب مدل مولفه‌ای ۸۰
۵-۳- مطالعه‌ی موردی ۸۳
۵-۴- پیاده‌سازی ۸۶
۵-۵- بحث ۹۱
۵-۶- جمع‌بندی ۹۴
۶- نتیجه‌گیری ۹۶
۶-۱- مقدمه ۹۷
۶-۲- بازبینی تحقق اهداف ۹۷

۶-۳- نتیجه‌گیری ۹۸

۶-۴- پیشنهادها و کارهای آینده ۹۹

لیست مقاله‌های مستخرج از پایان‌نامه ۱۰۰

منابع و مراجع ۱۰۳

واژه‌نامه فارسی ۱۰۹

واژه‌نامه انگلیسی ۱۱۴

فهرست شکل‌ها

شکل ۲-۱ نارسایی آسودگی در مواجهه با تراکنش‌های وابسته ۴۶
شکل ۳-۱ تفاوت زمانی در دستیابی به معیارهای آسودگی و سکون ۵۷
شکل ۳-۲ پیکربندی سیستم ارسال پیام ۵۸
شکل ۳-۳ نمودار توالی از یک سناریوی رفتاری سیستم ۵۸
شکل ۳-٤ نمودار توالی یک سناریوی رفتاری سیستم که در آن تراکنش‌ها به‌صورت تودرتو تکرار می‌شوند ۶۰
شکل ٤-۱ چرخه‌ی اجرای نرم‌افزار در مراحل مختلف تکامل ۷۱
شکل ٤-۲ ارسال غیرمستقیم پیام بین دو مولفه با استفاده از اتصال‌گرها ۷۴
شکل ٤-۳ نمودار وضعیت سیستم در حین تکامل پویا با استفاده از روش پیشنهادی ۷۸
شکل ۵-۱ اجزای یک مولفه در Fractal ۸۱
شکل ۵-۲ نمایی از زبان توصیف معماری در Fractal ۸۳
شکل ۵-۳ پیکربندی سیستم ارسال پیام با استفاده از اتصال‌گرهای صریح در معماری ۸۴
شکل ۵-٤ پیکربندی و نمودار توالی سیستم وقتی پیام از Sender به (De)Compression ارسال می‌شود ۸۵
شکل ۵-۵ پیکربندی معماری سیستم ارسال پیام در Fractal ۸۶
شکل ۵-٦ نمودار توالی سیستم با دو تراکنش هم‌زمان فعال ۹۱
شکل ۵-۷ نمودار توالی از فعالیت همزمان دوتراکنش تودرتو در سیستم ۹۳

فهرست جدول‌ها

جدول ۱-۱ مراحل انجام تحقیق ۱۰
جدول ۳-۱ مقایسه‌ی معیارهای سکون و آسودگی..... ۶۷
جدول ۵-۱ مقایسه‌ی روش پیشنهادی، سکون و آسودگی ۹۴

چکیده

بسیاری از برنامه‌های کامپیوتری نیازمند اجرای مداوم و بدون‌وقفه هستند. این شکل از اجرا در مورد برنامه‌های حیاتی چون پردازشگرهای تراکنش‌های مالی، سوییچ‌های مخابراتی، سیستم‌های کنترل ترافیک هوایی، سیستم‌های امنیتی نظامی و دیگر سیستم‌های نیازمند به دسترس‌پذیری بالا، به‌مراتب برجسته‌تر و از اهمیت بالاتری برخوردار است. در ساده‌ترین حالت که به بروزرسانی ایستا مشهور است، ارتقاء و برطرف‌سازی خطای یک سیستم، نیازمند متوقف کردن سیستم، به‌روزرسانی و نهایتا راه‌اندازی مجدد آن می‌باشد که چنین راهکاری در مورد سیستم‌های حیاتی ذکرشده به‌هیچ وجه قابل‌قبول نیست، چراکه توقف سرویس‌دهی در این سیستم‌ها می‌تواند منجر به خسارتهایی غیرقابل جبران شود. لذا این دسته از سیستم‌ها نیازمند استفاده از روش دیگری به‌نام به‌روزرسانی پویا هستند که امکان به‌روزرسانی یک سیستم در حال اجرا را با حداقل وقفه‌ی ممکن در خدمت‌رسانی آن مهیا می‌سازد. بنابراین کاهش وقفه‌ی خدمت‌رسانی سیستم حین انجام به‌روزرسانی پویا و حفظ سازگاری سیستم در طول این فرایند از مسائل حیاتی این حوزه است.

راه‌کارهایی که تاکنون به‌منظور به‌روزرسانی پویای نرم‌افزارهای مبتنی بر مولفه ارائه شده‌اند، در مواردی چون تراکنش‌های تودرتو، حذف مولفه و عملیات برگشت‌پذیری چون رمزگذاری و فشرده‌سازی، نارسایی‌هایی به‌همراه دارند که علاوه بر تاخیر در به‌روزرسانی، وقفه‌ی بسیاری در خدمت‌رسانی سیستم ایجاد می‌کنند.

این تحقیق با بهره‌گیری از مدل معماری، به‌کارگیری صریح اتصال‌گرها، و استفاده‌ی همزمان از نسخه‌ی قدیمی و به‌روزشده‌ی یک مولفه، روشی جدید به‌منظور حفظ سازگاری در تکامل پویای نرم‌افزار مبتنی بر مولفه ارائه می‌کند. نتایج حاصل از پیاده‌سازی و ارزیابی روش پیشنهادی با استفاده از یک نمونه‌ی مطالعاتی، دلالت بر تسریع به‌روزرسانی، کاهش وقفه‌ی خدمت‌رسانی و افزایش ایمنی در تکامل پویای سیستم دارد.

کلمات کلیدی: نرم‌افزار مبتنی بر مولفه، بازپیکربندی پویا، بازپیکربندی ایمن، معماری نرم‌افزار.

۱- کلیات تحقیق

۱–۱– مقدمه

بسیاری از برنامه‌های کامپیوتری نیازمند اجرای مداوم و بدون‌وقفه[1] هستند. این شکل از اجرا در مورد برنامه‌های حیاتی[2] همچون پردازشگرهای تراکنش‌های مالی، سوییچ‌های مخابراتی، سیستم‌های کنترل ترافیک هوایی، سیستم‌های امنیتی نظامی و دیگر سیستم‌های نیازمند به دسترس‌پذیری بالا[3]، به‌مراتب برجسته‌تر و از اهمیت بالاتری برخوردار است.

از سویی دیگر، شرکت‌ها نیازمند ارتقاء نرم‌افزارهای خود به‌منظور برطرف‌کردن خطاها، افزایش بهره‌وری و توسعه‌ی کارایی آن‌ها هستند. در ساده‌ترین حالت که به‌روزرسانی ایستا[4] نام دارد، ارتقاء و برطرف‌سازی خطای یک سیستم، نیازمند متوقف کردن[5] سیستم، بروزرسانی و نهایتاً راه‌اندازی مجدد آن می‌باشد. پرواضح است که چنین راه‌کاری در مورد سیستم‌های حیاتی ذکرشده به‌هیچ وجه قابل‌قبول نیست، چراکه توقف سرویس‌دهی در این سیستم‌ها می‌تواند خساراتی غیرقابل جبران در بر داشته‌باشد. بنابر نتایج ارائه شده در [۱]، ۷۵ درصد از حدود ۶۰۰۰ بار قطعی برق در برنامه‌های نیازمند به دسترس‌پذیری بالا به‌دلیل عملیات نگهداری نرم‌افزاری و سخت‌افزاری آن‌ها بوده‌است. به‌عنوان مثال، یک سیستم کنترل ترافیک هوایی را در نظر بگیرید که به‌دلیل تعلیق چندثانیه‌ای در سرویس‌رسانی و قطع ارتباط با هواپیماهای موجود در حریم هوایی تعریف‌شده، می‌تواند منجر به سانحه‌ی هوایی و به‌خطر افتادن جان صدها مسافر این هواپیماها شود.

[1] nonstop
[2] mission critical application
[3] highly availability
[4] static updating
[5] shutdown

بنابراین، اگر چه به‌روزرسانی پویای این سیستم‌ها به‌سادگی امکان‌پذیر نیست، استفاده از راه‌کاری به‌منظور حداقل نمودن وقفه‌ی خدمت‌رسانی در حین به‌روزرسانی این سیستم‌ها امری ضروری است [٤-٢].

همچنین علاوه بر سیستم‌های حیاتی، بسیاری از سیستم‌های دیگر نیز وجود دارند که لزوماً نیازمند اجرای بدون‌وقفه نیستند، ولی می‌توانند از چنین قابلیتی سود ببرند. به‌طور مثال، یک کامپیوتر رومیزی[1] را درنظر بگیرید که بدون نیاز به راه‌اندازی مجدد سیستم‌عامل پس از هربار به‌روزرسانی، بتواند بصورت پویا به‌روزرسانی را اعمال کند.

در مجموع، به‌روزرسانی پویا به توسعه و فرایند نگه‌داری نرم‌افزار کمک بسیاری می‌کند. همچنین این قابلیت با پشتیبانی از متدولوژی اصلاح و ادامه[2] در توسعه‌ی نرم‌افزار، امکان برطرف‌سازی خطا را مادامیکه برنامه در وضعیت مشاهده‌ی خطا قرار دارد، فراهم می‌سازد.

۱-۲- تعریف مساله

جهت جلوگیری از ابهام خواننده در اصطلاحات به‌کار گرفته شده در این تحقیق، لازم به‌ذکر است که منظور از بازپیکربندی[3]، پیکربندی مجدد[4]، تکامل[5] و تغییر[6] در این تحقیق، به‌روزرسانی (اصلاح یا جایگزینی) یک مولفه[7] می‌باشد. همچنین منظور از تغییر پویا، یا زمان اجرای[8] یک مولفه، بروزرسانی آن مولفه در طول اجرای نرم‌افزار می‌باشد.

[1] desktop PC
[2] fix-and-continue
[3] reconfiguration
[4] reconfiguration
[5] evolution
[6] change
[7] component
[8] runtime

تاکنون راه‌کارهای مختلفی جهت انجام به‌روزرسانی پویا[1] ارائه شده‌است. یکی از رایج‌ترین روش‌ها، استفاده از سیستم‌های افزونه[2] در تحمل‌پذیری خطا یا توازن بار سرویس‌رسانی می‌باشد. از آن‌جا که این راه‌کار، سربار هزینه و پیچیدگی سیستم را به‌دنبال دارد، استفاده از آن به‌منظور بروزرسانی مقرون به‌صرفه نیست. با استفاده از یک روش چندمنظوره و ساده‌تر، می‌توان بروزرسانی نرم‌افزار را بدون نیاز به سخت‌افزارهای اضافی انجام داد. در ادامه به معرفی روش مبتنی بر نرم‌افزار به‌روزرسانی پویا می‌پردازیم.

تا کنون چندین مدل‌مولفه‌ای[3] با فراهم نمودن واسط‌های برنامه‌نویسی[4] لازم جهت ساخت یا حذف پویای مولفه و اتصال‌دهنده[5]، از پیکربندی پویای سیستم‌های مبتنی بر مولفه پشتیبانی نموده‌اند [٦و٧]. از مهمترین مسائل در طول انجام پیکربندی، حفظ یکپارچگی[6] و سازگاری[7] عناصر و اطلاعات سیستم در حال اجرا است. از آن‌جا که اغلب، حفظ سازگاری در طول پیکربندی به‌صورت بومی[8] در مدل‌های مولفه‌ای وجود ندارد، عمل پیکربندی پویا هم‌چنان با چالش‌هایی مواجه است. لذا لازم است تا یک سیستم در حال اجرا پیش از هرگونه تغییر، ابتدا در وضعیتی سازگار قرار گیرد؛ چراکه در غیر اینصورت، هرگونه تغییر در نرم‌افزار می‌تواند منجر به بروز ناسازگاری و نهایتاً خرابی عملکرد سیستم شود. جهت روشن‌تر شدن مسئله، بحث را با یک مثال ادامه می‌دهیم: مولفه‌ای را درنظر بگیرید که در ابتدای کار مقادیر را رمزگذاری[9] و پس از انجام یکسری پردازش‌ها، نیازمند رمزگشایی[10] اطلاعات

[1] dynamic updating
[2] redundant
[3] component model
[4] API
[5] connector
[6] integrity
[7] consistency
[8] native
[9] encrypt
[10] decrypt

رمز شده است. حال درصورتی‌که پس از رمزگذاری اطلاعات، در نتیجه‌ی به‌روزرسانی سیستم، الگوریتم رمز مذکور از نوع A به نوع B تغییر کند، پرواضح است که مولفه‌ی مذکور، دیگر قادر به رمزگشایی اطلاعات از قبل رمزشده با الگوریتم A نخواهد بود. لذا انجام به‌روزرسانی در یک زمان تصادفی می‌تواند منجر به ناسازگاری‌هایی شود که سیستم را در وضعیتی نادرست[1] قرار دهد.

یکی از راهکارهای ارائه شده به‌منظور حل این دسته از مسائل، انتظار تا اتمام کامل عملیات مولفه‌ی مذکور می‌باشد. نظر به اینکه چنین انتظاری می‌تواند خود منجر به بروز یکسری نیازمندی‌های دیگر جهت حفظ سازگاری شود، لذا این روش لزوماً راه‌گشا نخواهد بود. از سویی دیگر، در صورت ایمن بودن وضعیت سیستم، معمولاً به‌منظور انجام به‌روزرسانی، لازم است تا همه‌ی مولفه‌های مرتبط با تغییرات مورد نیاز غیرفعال[2] شوند که وقفه‌ی بسیاری در خدمت‌رسانی سیستم به‌همراه خواهد داشت. تا کنون تحقیقات بسیاری به‌منظور تشخیص وضعیتی از سیستم در حال اجرا که در آن امکان انجام تغییر در مولفه‌های سیستم، به‌صورت ایمن و سازگار وجود دارد انجام شده‌است که در نتیجه‌ی این تحقیقات برخی معیارهای توقف ایمن[3] ارائه شده‌اند. علی‌رغم گستردگی تحقیقات در این حوزه، راه‌کارهایی ارائه شده علاوه بر تحمیل وقفه‌ی بسیار، نارسایی در تراکنش‌های توزیع‌شده، تاخیر در به‌روزرسانی و در مواردی تنها بر حفظ سازگاری محلی[4] نرم‌افزار تمرکز دارند [۹و۸و۳].

[1] erroneous
[2] deactivate
[3] safe stopping criteria
[4] local consistency