ISSN: 2962-3545
Prosiding SAINTEK: Sains dan Teknologi Vol.2 No.1 Tahun 2023
Call for papers dan Seminar Nasional Sains dan Teknologi Ke-2 2023
Fakultas Teknik, Universitas Pelita Bangsa, Februari 2023

Keterlibatan Pengguna Dalam Pengembangan Perangkat Lunak Dan Keamanan Dalam Pengembangan Perangkat Lunak

User involvement in software development and Security in software development

Johanes Mula Febrian Sihombing
Program Studi Teknik Informatika, Fakultas Teknik, Universitas Pelita Bangsa
Email : johanesmula@mhs.pelitabangsa.ac.id

Abstract

Software development consists of stages that are interrelated with each other. The analysis and planning stage is the initial stage in software development. For beginner software developers, making diagrams as software models is something that can be done as an alternative in the development of applications and computer systems. Software development today is huge and small organizations encourage every organization to develop and handle in terms of development. Computer programs Preparing for Progress can be done by referring to CMMI (Capability Development Shows Integration) made by SEI (Computer-Based Program Planning). In this research, interface design for software modeling applications was carried out. Experimentation is often used as a way to further validate user requirements and to aid device development decision-making. Involve users in the development of software products beneficial to users and companies. Software engineering provides the means to define, implement, and verify security in software products. Software security engineering is done by following the life cycle of software development models or security capability maturation models. An essential requirement is the integration of user experience methods in Agile software development. Based on this, the development of positive user experience must be managed. Management in general as a combination of a goal, a strategy, and resources. When applied to UX (User Experience), user experience management consists of a UX goal, a UX strategy, and UX resources.

Keywords*: Software development; Software security engineering; CMMI, UX*

Abstrak

Pengembangan perangkat lunak terdiri dari tahapan-tahapan yang saling terkait satu sama lain. Tahapan analisis dan perancangan merupakan tahapan awal dalam pengembangan perangkat lunak. Bagi pengembang perangkat lunak pemula, pembuatan diagram sebagai pemodelan perangkat lunak merupakan hal yang dapat dilakukan sebagai alternatif dalam pengembangan aplikasi maupun sistem komputer. Pengembangan perangkat lunak saat ini sangat besar dan organisasi kecil mendorong setiap organisasi untuk mengembangkan dan mengontrol dalam hal pembangunan. Program komputer Mempersiapkan Kemajuan dapat dilakukan dengan mengacu pada CMMI (Capability Development Shows Integration) yang dibuat oleh SEI (Perencanaan Program Berbasis Komputer). Dalam penelitian ini dilakukan perancangan antarmuka untuk aplikasi pemodelan perangkat lunak. Eksperimen sering digunakan sebagai cara untuk terus memvalidasi kebutuhan pengguna dan untuk membantu pengambilan keputusan pengembangan perangkat lunak. Libatkan pengguna di Pengembangan produk perangkat lunak bermanfaat bagi pengguna dan perusahaan. Rekayasa keamanan perangkat lunak menyediakan sarana untuk mendefinisikan, mengimplementasikan, dan memverifikasi keamanan dalam perangkat lunak produk. Rekayasa keamanan perangkat lunak dilakukan dengan mengikuti siklus hidup pengembangan keamanan perangkat lunak model atau model kematangan kapabilitas keamanan. Persyaratan penting adalah integrasi metode pengalaman pengguna dalam pengembangan perangkat lunak Agile. Berdasarkan hal tersebut, pengembangan pengalaman pengguna yang positif harus dikelola. Manajemen umumnya merupakan kombinasi dari tujuan, strategi, dan sumber daya. Saat diterapkan ke UX (User Experience), manajemen pengalaman pengguna terdiri dari sasaran UX, strategi UX, dan sumber daya UX.

Kata kunci: Pengembangan Perangkat Lunak, Rekayasa Keamanan Perangkat Lunak, CMMI, UX.

Pendahuluan

Saat ini perkembangan serta kemajuan dari teknologi informasi dan komunikasi merupakan suatu teknologi yang bisa digunakan untuk mengolah data, termasuk memproses, mendapatkan, menyusun, menyimpan, mamanipulasi information dalam berbagai cara untuk mendapatkan informasi yang mungkin relevan serta akurat dan tepat waktu, yang dapat digunakan untuk keperluan pribadi, pendidikan, bisnis maupun pemerintahan dan merupakan informasi yang strategis untuk dapat mengambil keputusan [1]. Eksperimen dengan pengguna sering digunakan untuk memahami perilaku dan kebutuhan pengguna saat mereka berinteraksi dengan produk dan layanan. Pandangan para praktisi dalam peran yang berbeda dapat membantu perusahaan lebih memahami dan meningkatkan praktik pengembangan berbasis eksperimen. [2].

Teknologi Informasi (TI) menjadi lebih relevan, sebagai pendukung produk dan layanan, sebagai keuntungan bagi masyarakat akses langsung ke pasar global, dan ekspektasi tinggi terkait layanan yang diberikan. Oleh karena itu, organisasi strategi mempertimbangkan penggunaan internet dan sistem komputer dalam skala besar, dengan penekanan pada persaingan keunggulan dibandingkan pesaing. Para eksekutif melaporkan bahwa penggunaan TI menguntungkan pendapatan organisasi pertumbuhan. Dalam konteks ini, organisasi harus mengidentifikasi area penerapan teknologi dan mengenal penggunanya persyaratan, menggabungkan ketangkasan dan kualitas dalam perangkat lunak yang digunakan, secara bersamaan. Perangkat lunak adalah elemen yang memperkenalkan teknologi baru atau memberikan layanan dalam organisasi[3].

Pengembangan perangkat lunak yang aman dilakukan dengan menjalankan serangkaian kegiatan rekayasa keamanan bersama dengan proses pengembangan perangkat lunak. Konon ini dilakukan dengan mengikuti model siklus hidup pengembangan keamanan, atau implementasi model kematangan keamanan [4]. Rekayasa keamanan sistem dan perangkat lunak telah menjadi aspek bisnis yang penting karena organisasi sepenuhnya bergantung pada sistem berbasis komputer dan menginvestasikan sumber daya yang substansial dalam pemeliharaannya[5].

Fase yang paling penting dan penting dalam pengembangan perangkat lunak adalah analisis kebutuhan. Kualitas proyek perangkat lunak sangat terkait dengan analisis kebutuhan. ianalisis dari berbagai mempelajari bahwa keberhasilan proyek perangkat lunak terkait dengan kualitas kebutuhan, Meskipun persyaratan awal set diidentifikasi dan didokumentasikan dengan baik, persyaratan akan diubah sepanjang proses pengembangan perangkat lunak. Ada dampak pada perkiraan jadwal, anggaran, kesalahan, dan kinerja proyek yang telah selesai jika persyaratan berubah terus menerus selama proses pengembangan perangkat lunak[6].

Metode tangkas terdiri dari kerangka manajemen proyek generik untuk pendekatan rekayasa perangkat lunak yang menyediakan praktik pengembangan perangkat lunak tertentu, seperti pairing pemrograman[7].Proses pengembangan perangkat lunak memiliki beberapa tahapan yang dilakukan diantaranya adalah pengumpulan kebutuhan, analisis, perancangan, implementasi, pengujian dan pemeliharaan. Tahapan analisis merupakan tahapan dalam memodelkan masalah yang dihadapi dalam pengembangan perangkat lunak, sedangkan tahapan perancangan adalah tahapan dalam memodelkan solusi pengembangan perangkat lunak [8].

Kegiatan penggunaan perangkat lunak menjadi kompleks karena adanya kebutuhan partisipasi langsung pengguna dalam perangkat lunak pemodelan untuk menghasilkan solusi terintegrasi. Penggunaan Agile Software Development (ASD) metodologi dalam manajemen proyek memungkinkan rilis perangkat lunak kepada pengguna dengan cepat dengan mengurangi waktu antara desain dan penerapan, mempromosikan pengujian parsial dan pengiriman dengan ketangkasan yang lebih besar daripada proyek klasik metode manajemen. Di antara praktik yang diadopsi dalam ASD kami menemukan (i) komunikasi tatap muka, (ii) rapat perencanaan berulang dan retrospektif yang difasilitasi oleh tim lintas fungsi yang mengatur diri sendiri, dan (iii) Integrasi berkelanjutan dengan pengujian memfasilitasi iterasi dan rilis singkat[3].

Untuk memastikan bahwa perangkat lunak tidak hanya memenuhi kebutuhan bisnis tetapi juga juga pengguna, penggunaan UCD (User-Centered Design) memungkinkan pengembang untuk memahami kebutuhan nyata pengguna dan membuat solusi yang dianggap umum sebagai memiliki peningkatan kegunaan, kegunaan, dan kepuasan pengguna. Intensitas keterlibatan pengguna bervariasi, dari konsultasi mereka kebutuhan dan partisipasi dalam pengujian kegunaan, agar pengguna aktif berpartisipasi dalam proses desain[7].

Pendekatan lain adalah penggunaan model kematangan UX. Keuntungan menggunakan model tersebut adalah bahwa hal itu menentukan tingkat kematangan saat ini sebuah organisasi. Dengan demikian, kelemahannya dapat diketahui. Tapi yang menentukan faktor adalah dimensi mana yang dipetakan dalam model kematangan UX. Untuk contoh, model Manajemen Pengalaman Pengguna Total (TUXM). berisi elemen-elemen seperti tujuan UX, sistem desain terintegrasi, komunikasi strategis, peningkatan berkelanjutan, pengambilan keputusan berdasarkan fakta, dan tim desain tipe-T. Kegunaan Perusahaan Nielsen Maturity Model, di sisi lain, terdiri dari dimensi seperti sikap pengembang terhadap kegunaan, sikap manajemen terhadap kegunaan, peran praktisi kegunaan, metode kegunaan dan teknik, dan kegunaan strategis. Sekilas, itu terlihat bahwa model TUXM berisi dimensi yang disebut tujuan UX yang tidak ada dalam model Nielsen. Sebaliknya, model Nielsen lebih terfokus pada implementasi praktis. Pengujian yang cocok Model kematangan UX harus dilakukan sebelum penyebaran dan disesuaikan dengan kebutuhan organisasi[9].

Tujuan utama dari rekayasa keamanan perangkat lunak adalah untuk menyediakan sarana bagi pengembang perangkat lunak untuk mematuhi persyaratan jaminan keamanan. Tujuan umum dari rekayasa keamanan perangkat lunak adalah untuk mengatasi risiko keamanan perangkat lunak yang teridentifikasi yang sudah dalam fase pengembangan, memungkinkan penciptaan solusi keamanan yang efektif secara efisien [4]. Kerentanan keamanan perangkat lunak yang paling populer dan terkenal adalah desain masalah, khususnya masalah desain arsitektur. Dari sistem perspektif pengembang, masalah keamanan perlu diidentifikasi lebih awal dalam langkah-langkah pengembangan pertama dan pada tingkat tertinggi, terutama pada tahap desain arsitektur, di mana semantiknya berada jernih. Ketika persyaratan keamanan ditentukan, aktivitas arsitektur dan desain dilakukan dengan menggunakan teknik dan alat pemodelan untuk kualitas yang lebih tinggi dan pengembangan yang mulus. Integrasi fitur keamanan menggunakan pendekatan ini membutuhkan keahlian tinggi untuk mengembangkan arsitektur dan desain dan ketersediaan kedua pengetahuan aplikasi-domain-spesifik dan keahlian keamanan[5].

Penelitian menunjukkan bahwa pengembang dan pemrogram ragu untuk menggunakan banyak penganalisa kode yang ada. Ketika beberapa studi telah melibatkan pandangan programmer di evaluasi alat analisis keamanan setelah selesai. Perancang alat sering kali gagal memasukkan komponen manusia sebelum dan selama desain dan pengembangan alat. menghindari kesalahan ini, saat merancang sistem yang dikenal sebagai VulIntel (Kerentanan IntelliSenser) untuk membantu pemrogram menulis lebih banyak kode aman, peneliti pekerjaan ini menjalankan keduanya studi formatif dan studi kegunaan untuk mendapatkan umpan balik dari programmer tentang pandangan dan harapan mereka terhadap alat[10]. Oleh karena itu, pengembang ini sistem perlu "merancang untuk keamanan". Ini termasuk mendefinisikan struktur sistem saat ini, yaitu arsitektur sistem, menemukan risiko abstrak dan kerentanan konkret, dan menerapkan tindakan pencegahan yang tepat untuk mengurangi risiko dan kerentanan untuk memenuhi persyaratan keamanan sistem ini[5].

Namun, bagaimana dan di mana melibatkan pengguna dan bagaimana menjalankan eksperimen dengan mereka sering dibentuk oleh konteks perusahaan, praktik yang ada dan berbagai perspektif dan sikap praktisi. Saat ini, tidak banyak penelitian yang menyelidiki realitas masalah ini di perusahaan pengembangan perangkat lunak. Dengan demikian, jurnal ini menyajikan studi survei yang dilakukan dengan perusahaan perangkat lunak untuk menyelidiki status development perangkat lunak yang ada dan praktik keterlibatan pengguna, perspektif para praktisi dalam melibatkan pengguna dalam pengembangan produk atau layanan mereka, dan bagaimana eksperimen perangkat lunak saat ini dipahami [2].

Metode Penelitian

Untuk mendapatkan pemahaman tentang pengembangan perangkat lunak yang sedang berlangsung dan aktivitas keterlibatan pengguna, serta eksperimen yang melibatkan pengguna di organisasi pengembangan perangkat lunak, dan untuk meningkatkan pemahaman tentang penggunaan kegiatan rekayasa keamanan dalam konteks pengembangan perangkat lunak tangkas, jurnal ini dimulai dengan mencari referensi penelitian dan membuat literature review. Lalu, Literatur yang dibahas dalam makalah ini dikumpulkan dari ScienceDirect, Google Scholar dan Sinta(Science and Technology Index). ScienceDirect, Google Scholar dan Sinta(Science and Technology Index) menyediakan abstrak semua publikasi diindeks, selain informasi tambahan, termasuk jumlah kutipan. Makalah yang terkait dengan penelitian ini disaring menggunakan literature review. Relevansi dan kualitas makalah yang dikumpulkan adalah dipastikan dengan mendefinisikan subjek dan kata kunci oleh karena itu analisis konten kualitatif dan analisis kuantitatif dilakukan pada kerangka penelitian yang ditampilkan.

Hasil dan Pembahasan

Untuk mengidentifikasi bagaimana perusahaan perangkat lunak melibatkan pengguna dalam praktik pengembangan, hasil dari jurnal ini diambil dari beberapa referensi penelitian, hasil tersebut dikumpulan dari ScienceDirect, Google Scholar, dan Sinta(Science and Technology Index). Bagian ini juga mencakup informasi dasar tentang organisasi mereka, sertifikasi, dan area application di mana keamanan telah menjadi perhatian dalam proyek perangkat lunak.

Pengembangan perangkat lunak tangkas muncul pada akhir 1990-an dan dianggap sebagai respons terhadap kegagalan perangkat lunak berbasis perencanaan yang ada.proses pengembangan, seperti Waterfall dan Rasional Unified Process untuk mengakomodasi sifat yang sangat fluktuatif persyaratan untuk proyek pengembangan perangkat lunak. Kritik umum terhadap metode ini adalah bahwa siklus hidup pengiriman perangkat lunak jauh lebih lambat daripada laju perubahan dalam domain masalah Misalnya, iterasi tipikal dalam Rational Unified Process adalah antara enam dan dua belas bulan, selama waktu itu, persyaratan untuk proyek atau teknologi yang tersedia di pasar mungkin telah banyak berubah. Pendukung pendekatan gesit untuk pengembangan perangkat lunak, malah menganjurkan model proses yang didasarkan pada berkelanjutan meninjau kemajuan dan persyaratan melalui kerjasama erat yang berkelanjutan dengan pelanggan[11].

Pendekatan lain adalah penggunaan model kematangan UX. Keuntungan menggunakan model tersebut adalah bahwa hal itu menentukan tingkat kematangan saat ini sebuah organisasi. Dengan demikian, kelemahannya dapat diketahui. Tapi yang menentukan faktor adalah dimensi mana yang dipetakan dalam model kematangan UX. Untuk contoh, model Manajemen Pengalaman Pengguna Total (TUXM). berisi elemen-elemen seperti tujuan UX, sistem desain terintegrasi, komunikasi strategis, peningkatan berkelanjutan, pengambilan keputusan berdasarkan fakta, dan tim desain tipe-T. Kegunaan Perusahaan Nielsen Maturity Model, di sisi lain, terdiri dari dimensi seperti sikap pengembang terhadap kegunaan, sikap manajemen terhadap kegunaan, peran praktisi kegunaan, metode kegunaan dan teknik, dan kegunaan strategis. Sekilas, itu terlihat bahwa model TUXM berisi dimensi yang disebut tujuan UX yang tidak ada dalam model Nielsen. Sebaliknya, model Nielsen lebih terfokus pada implementasi praktis. Pengujian yang cocok Model kematangan UX harus dilakukan sebelum penyebaran dan disesuaikan dengan kebutuhan organisasi[9].

Pendekatan pengembangan lain yang serupa dengan yang dilaporkan mengenai hal ini studi adalah Lean UX, yang didasarkan pada metode tangkas (biasanya Scrum), Design Thinking dan Lean Startup, fokus pada integrasi proses desain dengan pengembangan produk dan prinsip penggunaan berasal dari tiga pilarnya (misalnya, tim lintas fungsi dari Design Pikirkan, izin untuk gagal dari Lean Startup, dan keluar " bisnis yang dapat disampaikan " dari Agile). Model serupa lainnya termasuk Discovery By Design yang disebutkan di atas, serta Converge dan InnoDev; yang menyarankan pendekatan pengembangan perangkat lunak itu menggabungkan Agile, Design Thinking, dan Lean Startup. Penemuan Dengan Desain dan Converge

memiliki proses perkembangan yang ditandai dengan bobot penggunaan Pemikiran Desain (DX) selama fase awal mereka, diikuti dengan membangun siklus pengukuran-belajar dan memutar poin keputusan; dan didukung oleh data empiris yang dikumpulkan dari kasus industri dan sarjana tim pengembangan, masing-masing. InnoDev serupa, menyediakan fase pelingkupan di awal prosesnya; meskipun secara eksplisit menggunakan Scrum dan belum didukung oleh contoh kasus nyata[7].

Perusahaan yang menjadi fokus penelitian ini adalah perusahaan multi nasional yang besar yang mengembangkan produk di bidang avionik. Perusahaan terlibat dalam sejumlah proyek tentang desain dan pengembangan sistem safety-critical, yang terdiri dari perangkat keras dan perangkat lunak. Sebagai dibahas di atas, perusahaan telah mulai bereksperimen dengan penggunaan elemen proses Scrum dan praktik gesit lainnya. Selama ini periode, peneliti diundang untuk melakukan wawancara dengan jumlah karyawan perusahaan yang terlibat dalam hal ini proses transisi. Tujuan dari penelitian ini adalah untuk mengeksplorasi dan memahami penerapan pengembangan perangkat lunak tangkas untuk pengembangan perangkat lunak untuk sistem keselamatan-kritis dari perspektif praktisi. Studi ini berusaha untuk mengidentifikasi keduanya: manfaat yang diakui oleh praktisi dalam menggunakan metode dan praktik tangkas dalam konteks ini dan tantangan dan keterbatasan yang dialami. Kami melakukan serangkaian wawancara dengan praktisi di perusahaan[11].

Untuk memulainya, responden ditanya kegiatan pengembangan mana yang paling melibatkan pengguna. Tiga perusahaan menyatakan bahwa pengguna lebih terlibat dalam menentukan persyaratan – 71%, 75% dan 79% perjanjian di Perusahaan A, B, dan D masing-masing. Ini tidak mengherankan karena persyaratan untuk suatu produk atau layanan sering kali berasal dari pengguna dan pelanggan yang ada atau potensial. Perusahaan C, di sisi lain, menyatakan bahwa pengujian adalah kegiatan di mana mereka paling sering melibatkan pengguna (57% setuju). Secara total, hasil agregat menunjukkan bahwa sementara menentukan persyaratan adalah aktivitas di mana pengguna paling banyak terlibat, dengan 72% perjanjian, implementasi perangkat lunak adalah di mana pengguna terlibat paling sedikit, dengan 33% persetujuan. menunjukkan rata-rata tanggapan dari masing-masing perusahaan atas pernyataan mengenai keterlibatan pengguna, yang merupakan pertanyaan 7 dari survei. Secara keseluruhan, praktisi menyatakan bahwa mereka tahu siapa yang menggunakan perangkat lunak yang mereka kontribusikan di tempat kerja dan bahwa mereka memiliki cukup informasi tentang mereka[2].

Gambar 1 Pernyataan keterlibatan pengguna

Responden umumnya berpendidikan tinggi dan sangat berpengalaman dalam rekayasa perangkat lunak. Sementara sekitar lima persen melaporkan tidak ada pengalaman dalam pengembangan perangkat lunak, sebanyak 77% melaporkan enam tahun atau lebih. Kira-kira sekitar setengah dari responden memiliki gelar master, sedikit di bawah sepertiga memiliki gelar sarjana, dan sekitar lima belas persen memiliki gelar doktor.

Responden bekerja di berbagai organisasi. Sekitar 39% bekerja di organisasi dengan kurang dari lima puluh karyawan, sekitar 21% di organidengan 50 hingga 250 karyawan , dan sekitar 35% di organisasi dengan lebih dari 250 karyawan. Sekitar lima persen lebih memilih untuk tidak mengungkapkan informasi ini. Kisaran ini mungkin mencerminkan struktur industri perangkat lunak Finlandia saat ini. Hal yang sama berlaku untuk jenis perangkat lunak yang diproduksi. Sebagian besar (69%) responden mengembangkan aplikasi web dan cloud. Sisanya sebagian besar bekerja dengan aplikasi desktop dan klien-server, perangkat lunak tertanam, dan aplikasi seluler. Beberapa domain khusus juga diwakili, termasuk video game, kartu pintar, dan konsultasi. Sasaran utama survei adalah karyawan penuh waktu yang terlibat dalam peran pengembangan perangkat lunak teknis. Hasil penargetan ini adalah ditunjukkan pada Tabel 1: sebanyak 87% responden pernah langsung bekerja dalam sebuah proyek pengembangan perangkat lunak dengan pertimbangan keamanan yang dikelola menggunakan metodologi agile. Relevansi evaluasi Sebagian besar pertanyaan penelitian didasarkan pada angka tersebut, juga dengan mempertimbangkan tingkat pendidikan yang relatif tinggi dan pengalaman kerja yang panjang dari responden[4].

Tabel 1 Peran Proyek

No	Role	Share (%)
1	Pengembang	40
2	Arsitek	21
3	Serum master atau pemimpin tim	11
4	Manajer proyek	10
5	Eksekutif	8
6	Tidak ada proyek	5
7	Spesialis keamanan	3
8	Pemilik produk	2
9	Penguji	0

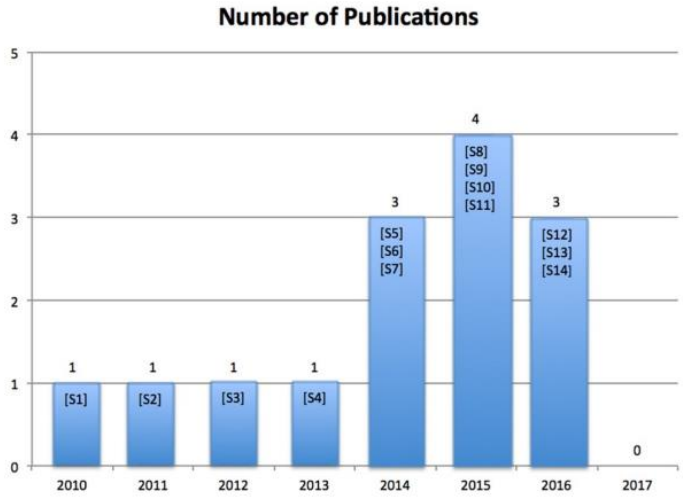
Responden juga ditanyai tentang metodologi tangkas tertentu yang digunakan dalam proyek dengan kendala keamanan yang mereka kerjakan. Itu daftar standar metodologi yang diberikan kepada responden adalah dirakit berdasarkan survei industri baru-baru ini. Hasil yang dirangkum dalam Tabel 2 tidak menunjukkan kejutan: Scrum dan variannya saat ini merupakan metodologi agile yang paling populer di kalangan responden. Di dalam berbeda dengan survei sebelumnya, tidak ada responden yang melaporkan menggunakan XP atau metode turunan dari XP. Beberapa responden melaporkan tidak menggunakan metodologi tangkas tertentu, atau menolak untuk menamainya. Khususnya, semua responden ini melaporkan menggunakan mis. iterasi dan jaminan simpanan, keduanya praktik utama dalam metodologi tangkas[4].

Tabel 2 Metodologi pengembangan.

No	Metodologi	Share (%)
1	Scrum	30
2	Scrum-Kanban	28
3	Kustom gesit	17
4	Kanban	14
5	Lainnya	3
6	Tidak Tahu	5
7	Tidak ada jawaban	3

Untuk menunjukkan hasil penelitian ini, sintesa naratif adalah digunakan. Sintesis naratif adalah metode yang umum digunakan untuk mensintesis penelitian dengan ukuran yang sama, termasuk tinjauan sistematis, dengan penggunaan pendekatan naratif (berlawanan dengan statistik) untuk meringkas temuan dari

pembelajaran. Mendukung narasi ini, kami mengklasifikasikan dan menyajikan hasilnya dalam urutan kronologis. Grafik pertama pada Gambar. 2 menyajikan distribusi publikasi setiap tahun. Alasan untuk ini dapat memiliki beberapa sumber, di antaranya dapat kami sebutkan: pada saat itu dimungkinkan tidak memiliki pekerjaan yang cukup di lapangan untuk memungkinkan tinjauan sistematis, atau istilah atau konsep lain mungkin telah digunakan pada saat itu dan istilah pencarian yang digunakan dalam penelitian ini tidak mengidentifikasi kertas, karena kosakata yang digunakan beberapa tahun yang lalu mungkin berbeda[12].



Gambar 2 Distribusi studi sekunder yang diterbitkan ditinjau setiap tahun.

Kami menganalisis data menurut prosedur analisis isi dengan Krippendorff, disusun menurut langkah-langkah berikut: pengorganisasian dan pra-analisis, membaca dan mengkategorikan, dan merekam hasilnya. Menggunakan Atlas. kami pertama-tama membaca kumpulan data, mengekstrak bagian teks dan menandainya sebagai kode. Kode ini telah ditinjau dan dikompilasi menjadi kode yang lebih besar, membentuk kategori. Proses iteratif ini dilakukan oleh dua orang peneliti dan ditinjau terus menerus oleh dua peneliti senior untuk mengurangi apapun keterbatasan atau bias dalam analisis ini. Kami menggunakan hasil analisis ini untuk mempersiapkan kutipan deskriptif pendekatan tim gabungan secara keseluruhan. Sementara beberapa kategori adalah dibuat eksplisit, kami agregat sebagian besar untuk lebih menggambarkan beberapa konsep yang lebih umum dari pendekatan gabungan. Yang mengatakan, kita bisa memetakan prosedur pengumpulan khusus untuk setiap temuan kami. Misalnya, kami menggabungkan data yang diperoleh dari pengamatan dengan data bengkel untuk membentuk gambaran lengkap tentang proses kerja tim. Jurusan kami temuannya adalah: deskripsi dan tanggung jawab Peran yang membuat meningkatkan tim; Alur Kerja Pengembangan yang digunakan oleh peserta; Pengaruh Milestones terhadap proses pembangunan; dan Tim Efek dari pendekatan gabungan, terkait dengan pemikiran mereka dan keseluruhan "merasa"[7].

Pada bagian berikut, kami menyajikan sorotan yang ditemukan dalam studi yang disertakan. Pendekatan ini berpotensi sesuai untuk mengelola proses UX saat mereka mengintegrasikan metode UX ke dalamnya pengembangan perangkat lunak yang gesit. Pendekatan yang paling sering diidentifikasi adalah 'Desain UCD Muka'. Tim UCD kedua akan ditambahkan ke tim pengembangan sebenarnya. Tim ini bekerja di jalur paralel, selalu satu iterasi di depan iterasi pengembangan. Di jalur paralel ini, prototipe biasanya dibuat dalam berbagai bentuk. Prototipe ini kemudian diserahkan kepada tim pengembangan dan dikembangkan pada iterasi iterasi berikutnya[9].

Konfigurasi keamanan aplikasi, persyaratan desain, dan kasus penyalahgunaan atau penyalahgunaan adalah aktivitas desain keamanan yang paling sering digunakan. Ini semua sangat mirip dengan aktivitas yang dilakukan di sebagian besar proyek pengembangan perangkat lunak tangkas pada umumnya: penggunaan yang tinggi kemungkinan dijelaskan oleh pengembang yang merasa sangat wajar untuk menambahkan rasa keamanan ke dalam aktivitas umum ini. Pengamatan ini penting karena regulasi umumnya dianggap sebagai sumber persyaratan keamanan yang signifikan dalam pengembangan perangkat lunak. Masalah mendasar dengan standar keamanan perangkat lunak, seperti Common Criteria, adalah kelalaian proses bisnis yang bisa dibilang harus menjadi pendorong utama untuk rekayasa keamanan.

Rekayasa kegunaan didefinisikan sebagai "suatu proses di mana kegunaan suatu produk dinyatakan secara kuantitatif, dan dalam pertama. Kemudian, saat produk dibuat, pengujian dilakukan untuk melihat apakah tingkat kegunaan yang direncanakan telah tercapai". Tujuan Rekayasa kegunaan adalah untuk rekayasa untuk perbaikan dan sifatnya adalah tercermin dalam pengembangan iteratif dengan siklus desain ulang nilai desain. Rekayasa kegunaan mendefinisikan lima karakteristik kegunaan utama yang terkait dengan masalah seperti kemampuan belajar sistem, efisiensi, daya ingat, kesalahan, dan kepuasan pengguna[12].

Persyaratan yang tidak pasti adalah alasan utama untuk mengubah persyaratan dalam pengembangan proyek perangkat lunak yang digariskan, yaitu persyaratan dapat berubah secara tidak terduga kemudian berdampak pada pengembangan proyek perangkat lunak melalui pengerjaan ulang. Ini sedang dianalisis bahwa ketidakpastian kebutuhan memiliki potensi ancaman atau faktor risiko utama dalam pengembangan proyek perangkat lunak sebagaimana diuraikan. Oleh karena itu, penting untuk menangani ketidakpastian persyaratan ini selama proses pengembangan proyek dan apakah akan menghapusnya ketidakpastian ini atau meminimalkan dampaknya. Juga telah diidentifikasi bahwa ketidakpastian kebutuhan sebagai elemen mendasar dalam perencanaan strategis bekerja untuk memenuhi perkiraan waktu. Itu wajar untuk proyek perangkat lunak untuk meningkatkan anggaran dalam hal biaya, waktu pengiriman, dan kualitas proyek melalui penanganan persyaratan yang tidak pasti dengan tepat disarankan[6].

Ini menegaskan bahwa masuk akal untuk menggunakan varian model yang memiliki struktur yang disesuaikan dengan tertentu teknik prediksi, karena beberapa varian untuk beberapa teknik tampaknya tidak berguna meskipun teknik tertentu tidak berguna umumnya dianggap berkinerja baik. Jadi, kami melakukan analisis serupa tetapi hanya melibatkan varian terbaik untuk masing-masing teknik dipilih nilai 'perbedaan'. Ini memungkinkan menghindari situasi di mana model tertentu varian dibandingkan dengan varian lain yang dibuat dengan buruk[13],[14],[15].

Dampak keamanan dari persyaratan desain dan sebagai kasus penyalahgunaan atau penyalahgunaan juga dianggap tinggi, menambah kegunaan yang dirasakan. Dampak keamanan dari kegiatan fase desain lainnya dianggap jauh lebih rendah, membuat kedua kegiatan ini berbeda. Berbeda dengan aktivitas desain perangkat lunak umum, pemodelan ancaman dan analisis permukaan serangan secara eksklusif merupakan tugas rekayasa keamanan. Ini adalah penjelasan yang mungkin untuk penggunaannya yang relatif jarang[16].

Kesimpulan

Manfaat pengembangan berbasis eksperimen dan pengambilan keputusan berbasis data diakui oleh banyak perusahaan pengembangan perangkat lunak dan akademisi yang sukses. Pelopor teknologi besar diketahui menjalankan hingga ratusan eksperimen sekaligus di platform eksperimen mereka sendiri. Namun, di banyak organisasi lain, terutama di mana budaya eksperimen belum sepenuhnya rusak, kurangnya pemahaman tentang sumber daya dan kemampuan dapat menghambat transisi ke eksperimen berkelanjutan. Dalam pekerjaan di masa depan, mereplikasi studi kami dalam konteks dan populasi perusahaan yang berbeda akan menjadi penting untuk mendapatkan data tambahan untuk mengevaluasi dan meningkatkan temuan kami.

Yang penting, penelitian ini telah menunjukkan bahwa ada kebutuhan untuk beradaptasi pengembangan perangkat lunak tangkas agar sesuai dengan batasan perangkat lunak pengembangan untuk sistem kritis keselamatan dan menyelidiki yang spesifik tantangan secara rinci. Secara khusus, ada kebutuhan untuk

memahami caranya pengembangan perangkat lunak tangkas dapat diskalakan agar sesuai dengan skala besar dan kompleks upaya rekayasa sistem yang terdiri dari berbagai upaya pengembangan yang mencakup komponen perangkat lunak dan perangkat keras proyek dapat berlangsung puluhan tahun. Akhirnya, pertanyaan-pertanyaan ini mencerminkan kebutuhan untuk lebih baik mengoordinasikan tempo pengembangan sistem keselamatan-kritis dan itu diasumsikan oleh pengembangan perangkat lunak tangkas. Filosofi tangkas adalah untuk mengakomodasi perubahan proyek pengembangan perangkat lunak yang terus menerus, cepat, bersamaan, karena tekanan eksternal yang tidak dapat dihindari. Kompleksitas yang diciptakan oleh perubahan ini kemudian dikurangi melalui disiplin penerapan kombinasi alat dan metode. di sisi lain, filosofi dalam pengembangan perangkat lunak untuk sistem keamanan kritis adalah sengaja membatasi pilihan untuk (dan tingkat) perubahan untuk mempertahankan ketertelusuran artefak. Menerapkan pengembangan perangkat lunak yang gesit untuk keselamatan sistem kritis, oleh karena itu, memerlukan pengembangan alat dan metode yang memberikan standar berkelanjutan yang sama ketertelusuran. Survei, yang dilakukan di antara praktisi perangkat lunak, memberikan pandangan positif tentang tingkat profesionalisme dan kemampuan keamanan responden. Responden memberikan penampilan menganggap serius pekerjaan keamanan, dan mereka benar-benar khawatir tentang keamanan produk perangkat lunak yang mereka kembangkan.

Daftar Rujukan

[1] J. Ilmiah and K. Grafis, "Analisis Desain Software Process Improvement Untuk Organisasi Pengembang Perangkat Lunak Skala Usaha Kecil," vol. 15, no. 1, pp. 191–195, 2022, [Online]. Available: http://journal.stekom.ac.id/index.php/pixel■page191

[2] S. Yaman, F. Fagerholm, M. Munezero, T. Männistö, and T. Mikkonen, "Patterns of user involvement in experiment-driven software development," *Inf Softw Technol*, vol. 120, Apr. 2020, doi: 10.1016/j.infsof.2019.106244.

[3] J. C. Pereira and R. de F. S. M. Russo, "Design thinking integrated in agile software development: A systematic literature review," in *Procedia Computer Science*, 2018, vol. 138, pp. 775–782. doi: 10.1016/j.procs.2018.10.101.

[4] K. Rindell, J. Ruohonen, J. Holvitie, S. Hyrynsalmi, and V. Leppänen, "Security in agile software development: A practitioner survey," *Inf Softw Technol*, vol. 131, Mar. 2021, doi: 10.1016/j.infsof.2020.106488.

[5] B. Hamid and D. Weber, "Engineering secure systems: Models, patterns and empirical validation," *Comput Secur*, vol. 77, pp. 315–348, Aug. 2018, doi: 10.1016/j.cose.2018.03.016.

[6] M. Haleem, M. F. Farooqui, and M. Faisal, "Cognitive impact validation of requirement uncertainty in software project development," *International Journal of Cognitive Computing in Engineering*, vol. 2, pp. 1–11, Jun. 2021, doi: 10.1016/j.ijcce.2020.12.002.

[7] M. Zorzetti, I. Signoretti, L. Salerno, S. Marczak, and R. Bastos, "Improving Agile Software Development using User-Centered Design and Lean Startup," *Inf Softw Technol*, vol. 141, Jan. 2022, doi: 10.1016/j.infsof.2021.106718.

[8] E. R. Subhiyakto, Y. P. Astuti, and L. Umaroh, "KONSTELASI: Konvergensi Teknologi dan Sistem Informasi Perancangan User Interface Aplikasi Pemodelan Perangkat Lunak Menggunakan Metode User Centered Design."

[9] A. Hinderks, F. J. Domínguez Mayo, J. Thomaschewski, and M. J. Escalona, "Approaches to manage the user experience process in Agile software development: A systematic literature review," *Inf Softw Technol*, vol. 150, Oct. 2022, doi: 10.1016/j.infsof.2022.106957.

[10] IEEE Staff, *2019 SoutheastCon*. IEEE, 2019.

[11] G. Islam and T. Storer, "A case study of agile software development for safety-Critical systems projects," *Reliab Eng Syst Saf*, vol. 200, Aug. 2020, doi: 10.1016/j.ress.2020.106954.

[12] K. Curcio, R. Santana, S. Reinehr, and A. Malucelli, "Usability in agile software development: A tertiary study," *Computer Standards and Interfaces*, vol. 64. Elsevier B.V., pp. 61–77, May 01, 2019. doi: 10.1016/j.csi.2018.12.003.

[13] L. Radlinski, "Stability of user satisfaction prediction in software projects," in *Procedia Computer Science*, 2020, vol. 176, pp. 2394–2403. doi: 10.1016/j.procs.2020.09.308.

[14] R. E. Riantini, "Scrum to support application development project for online learning," pp. 58–64, 2021.

[15] M. T. Baldassarre, "Integrating security and privacy in HCD-scrum," ACM Int. Conf. Proceeding Ser., 2021, doi: 10.1145/3464385.3464746.

[16] J. Reichwein, S. Vogel, S. Schork, J. Dantan, A. Etienne, and A. Siadat, "ScienceDirect ScienceDirect ScienceDirect Design On Agile Development Methods Design On the the Applicability Applicability of Agile Development Methods to Design for for Additive Additive Manufacturing Manufacturing A new methodology to analyze the funct," Procedia CIRP, vol. 91, pp. 653–658, 2020, doi: 10.1016/j.procir.2020.03.112.