FRI-ONLINE-1-CCT1-06

RESEARCH OF SOME SPECIFICS OF MODERN ENVIRONMENTS FOR AUTOMATED SOFTWARE INSTALLATION AND IMPLEMENTATION[6]

Eng.Vladislav Hinkov, PhD student
Department of computing
"Angel Kanchev" University of Ruse
Phone: +359 884 260 685
E-mail: vhinkov@uni-ruse.bg

Prof. Georgi Krastev
Department of computing
"Angel Kanchev" University of Ruse
Phone: +359 82 888 672
E-mail: georg@uni-ruse.bg

***Abstract**: The paper reviews the types of modern environments for automated software installation and implementation, their characteristics and their application according to their different specifics. Modern environments for automated software research are increasingly entering the work of both large and small organizations. Issues such as security, quick and easy access to users, optimization of resources - hardware, human and financial, are the basis of their application. Guidelines are given for future theoretical and practical research.*

***Keywords:** Effectiveness, automated software installation, software mplementation, optimization of resource, cloud computing, docker, microsevices.*

***JEL Codes:** L86*

ВЪВЕДЕНИЕ

Внедряването на софтуер е сложен процес, който се състои в предоставянето на софтуер за използване и след това поддържането му в експлоатация. Засяга редица взаимосвързани дейности като освобождаване, инсталиране, активиране, актуализиране и др. Внедряването трябва да управлява огромен брой разнородни устройства и мрежови връзки и множество компоненти и софтуерни версии. Изправено пред мобилност, прекъсвания и вариации в наличността и качеството на ресурсите и услугите, внедряването също трябва да се променя и динамично да се адаптира, за да задоволи трайно ограниченията и изискванията (т.е. да запази очакваните свойства): трябва да се справи с динамиката и откритост както на мрежата от хостове (напр. поява и изчезване на устройства), така и на внедрената софтуерна система (например нови компоненти за интегриране, премахване или преместване на съществуващи).

Целта на работата е да се направи проучване на водещите технологии за автоматично инсталиране и внедраване на софтуер, като се изседва същността на внедряването в контекста на големи корпоративни мрежи. Да се дадат насоки за бъдещи изседвания.

ИЗЛОЖЕНИЕ

Изясняване значението на процеса „Software deployment"

Според Роджър С. Пресмен (Pressman, R. S., 2005), внедряването на софтуер представляват всички дейности, които правят една софтуерна система достъпна за използване.

[6] Докладът е представен на сесия FRI-ONLINE-1-CCT1 на 13 ноември 2020 г. с оригинално заглавие на български език: ИЗСЛЕДВАНЕ НА НЯКОИ СПЕЦИФИКИ НА СЪВРЕМЕННИТЕ СРЕДИ ЗА АВТОМАТИЗИРАНО ИНСТАЛИРАНЕ И ИЗПЪЛНЕНИЕ НА СОФТУЕР

Според Рийс-Картър Стивън (Rees-Carter S., 2018) общият процес на внедряване се състои от няколко взаимосвързани дейности с възможни преходи между тях. Тези дейности могат да се извършват от страна на производителя или от страна на потребителя или и двете. Тъй като всяка софтуерна система е уникална, точните процеси или процедури във всяка дейност трудно могат да бъдат дефинирани. Следователно „внедряване“ трябва да се тълкува като общ процес, който трябва да бъде персонализиран в съответствие със специфични изисквания или характеристики.

Сложността и изменчивостта на софтуерните продукти насърчиха появата на специализирани роли за координиране и проектиране на процеса на внедряване. За настолни системи крайните потребители често се превръщат и в „разгръщащи софтуера“, когато инсталират софтуерен пакет на своята машина. Внедряването на корпоративен софтуер включва много повече роли и тези роли обикновено се променят с напредването на приложението от тестовата (предпроизводствена) към производствена среда.

Изследване внедряването на софтуер поредством облачни технологии

Интеграцията с облачни технологии, е система от инструменти и технологии, която свързва различни приложения, системи, хранилища и ИТ среди за обмен на данни и процеси в реално време. Веднъж комбинирани, данните и интегрираните облачни услуги могат да бъдат достъпни от множество устройства през мрежа или чрез интернет. Внедряването на софтуер посредством облачни технологии е създадено, за подобри свързаността и видимостта и в крайна сметка да оптимизира процесите на взаимодействие. Това е отговор на необходимостта от споделяне на данни между приложения, базирани на облак, и за обединяване на информационните компоненти. Присобовяването към облака може да включва създаване на интеграция от облак към облак, от облак към локална система или комбинация от двете. Интеграциите могат да адресират различни компоненти, като данни и приложения. Интеграцията на приложения свързва различни приложения и организира непрекъсната функционалност и оперативна съвместимост. Това е повече от споделяне на данни. Включва издаване на заявки и команди за задействане на събития или процеси, (Zhang, J., 2020). Всяко управлявано събитие е предварително дефинирано и задейства процеса на внедряване при поискване, когато бъде открито. Въз основа на сложността на събитията, ние разделяме събитията на две категории: примитивни събития и сложни събития. Примитивното събитие е отражение на контекста в определен момент, може да се обозначи като:

$$PrimitiveEvent = InternalEvent \cup ExternalEvent$$

$$InternalEvent = \sum_{i=1}^{m} \{< internal, expression, t >_i\}$$

$$ExternalEvent = \sum_{i=1}^{n} \{< external, expression, t >_i\}$$

където {internal} е набор от вътрешни контексти, докато {external} е набор от контексти на околната среда; {expression} е набор от атрибутни функции; и t е означението за време. Всяко примитивно вътрешно събитие е кортеж, означен като <internal, expression, t>. По същия начин всяко примитивно външно събитие е кортеж, означен като <external, expression, t>. PrimitiveEvent е съвкупността от примитивни събития, която съдържа всички вътрешни събития, т.е. InternalEvent и всички външни събития, т.е. ExternalEvent.

*Таблица 1.*показва някои примери за примитивни събития в медицинска апратура.

Сложното събитие е логическа или временна композиция от примитивни събития, обозначени като:

$ComplexEvent = (\{event\}, \{operator\}, \{attribute\})$,

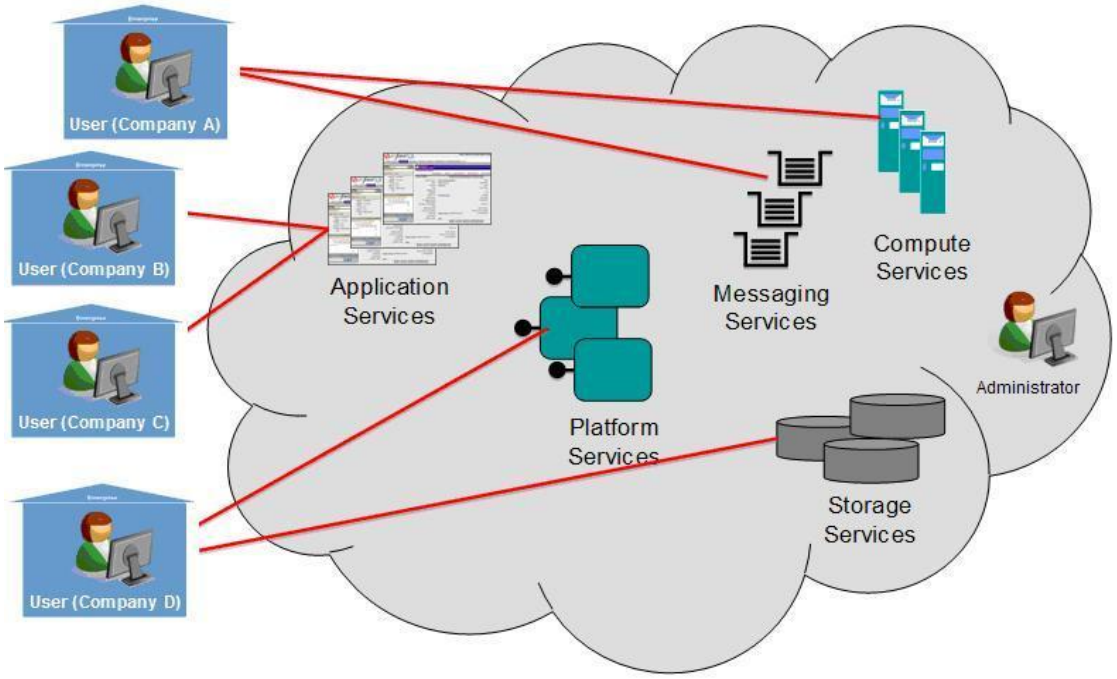
където {event} е съвкупността от събития, примитивни събития или сложни събития; {opertor} е съвкупността от логически и времеви оператори на композицията (напр. и, или, след това и т.н.); и {attribute} означава атрибути и ограничения на композицията.

Таблица 1. Примерни примитивни събития в медицинска апаратура.

Category	Context	Expression
Environment contexts	c_d: distance	Greater than 2000m
	c_h: heatbeat	$\leq$ 60 or $\geq$ 120 beats/min
System contexts	c_n: network delay	Greater than 0.3s
	c_c: CPU usage rate	Greater than 50%
	c_r: RAM usage rate	Greater than 50%

Видове облачи структури

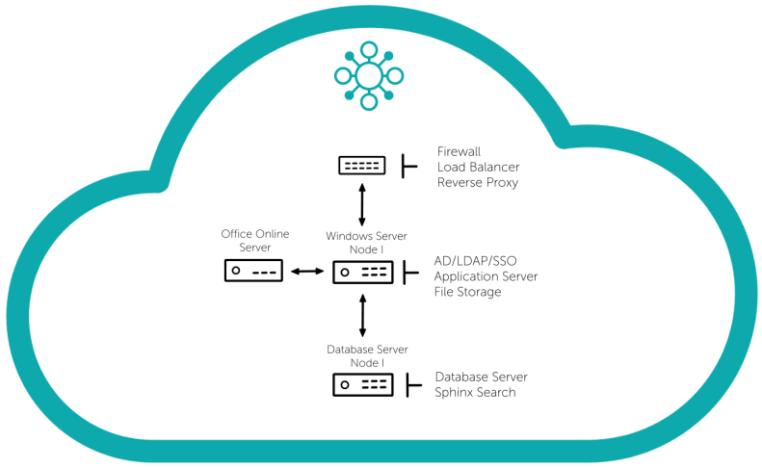
- Public Cloud е мястото, където външен доставчик предлага облачни услуги на множество клиенти.



Фиг. 1. Схема на публичен облак

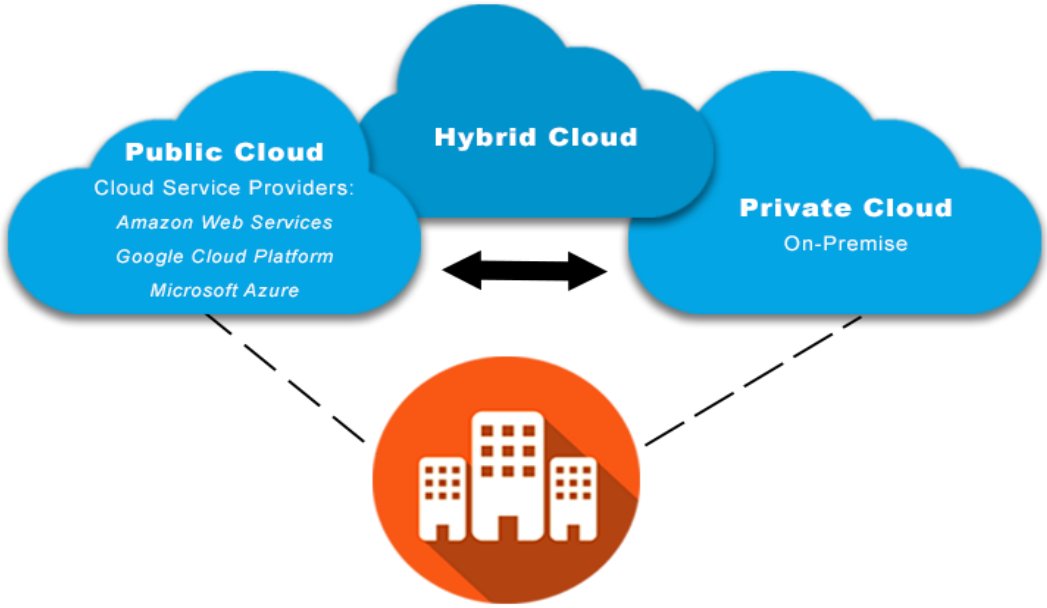
Основните доставчици на пулични облачни услуги са гиганти като : **Microsoft, Oracle, VM Ware, AWS, Google, SAP, IBM**

- Private Cloud е мястото, където облачните услуги са предназначени за един клиент. Инфраструктурата може да бъде собственост на самата организация или на външен доставчик на облак.



Фиг. 2. Примерна схема на частен облак

- Хибридният облак е мястото, където потребителят има комбинация от публичен и частен облак. Често те започват с Private Cloud и след това се разширяват до Public Cloud за допълнителна мащабируемост.



Фиг. 3. Хибридна облачна структура

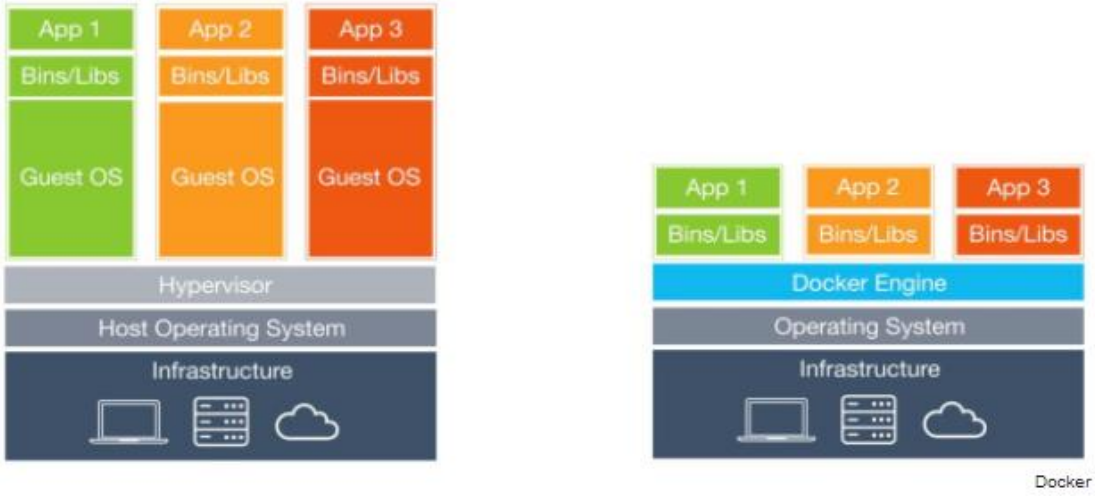
Характеристики на облачните услуги

- Използване при поискване: използването може да се самообслужва от потребителя и да се автоматизира, като не се изисква допълнителна работа от доставчика.
- Универсален достъп: може да бъде достъпен от всяко място с всяко устройство.
- Многостранност: услугите могат да се предоставят сигурно на множество потребители чрез една и съща споделена инфраструктура.
- Еластичност: ресурсите могат бързо да се мащабират или намаляват при поискване. Заедно с премерената употреба, това е голяма точка за продажба пред традиционните локални решения.
- Измерено използване: Използването на потребителски ресурси може да бъде проследено и таксувано.
- Устойчивост: услугите могат автоматично да бъдат отказани до определен ресурс.

Изследване инсталирането и внедряването на софтуер чрез използване на докер контейнери и микроуслуги

Според (Potdara A**.** M., 2020), контейнеризацията е технология, която съчетава приложението, свързаните зависимости и организираните системни библиотеки за изграждане под формата на контейнер. Приложенията, които са изградени и организирани, могат да се изпълняват и разгръщат като контейнер.

Docker е софтуерна платформа за изграждане на приложения, базирани на контейнери в малки и леки среди за изпълнение, които използват споделено ядрото на операционната система, но иначе се изпълняват изолирано една от друга. Една от целите на съвременното разработване на софтуер е да държи приложенията на един и същ хост или клъстер изолирани един от друг, за да не си пречат неправомерно работата или поддръжката. Това може да бъде трудно благодарение на пакетите, библиотеките и други софтуерни компоненти, необходими за тяхното стартиране. Едно от решенията на този проблем са виртуалните машини, които поддържат приложенията на един и същ хардуер напълно отделни и намаляват до минимум конфликтите между софтуерните компоненти и конкуренцията за хардуерни ресурси. Но виртуалните машини са обемисти - всяка се нуждае от собствена операционна система, така че обикновено е с размер на гигабайта - и е трудно да се поддържа и надгражда. Контейнерите, напротив, изолират изпълняващите среди на приложенията един от друг, но споделят основното ядро на ОС. Те обикновено се измерват в мегабайта, използват много по-малко ресурси от виртуалните машини и стартират почти веднага. Контейнерите осигуряват високоефективен механизъм за комбиниране на софтуерни компоненти във видовете стекове за приложения и услуги, необходими в съвременното предприятие, и за поддържане на тези софтуерни компоненти актуализирани и поддържани.



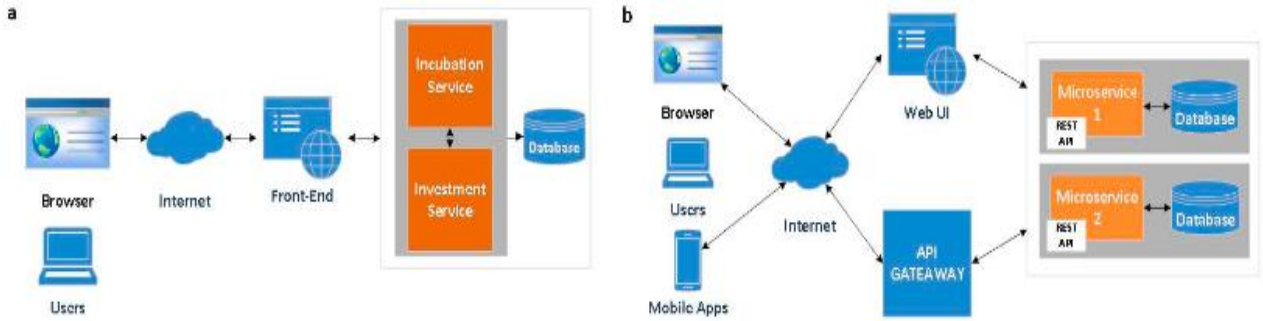
Фиг. 4.. Как се подреждат стековете за виртуализация и контейнерна инфраструктура.

Микроуслуги: (Butzin, B., 2016) описват микроуслугите по следния начин „архитектурният стил на микроуслугата е подход, използван за разработване на едно приложение като набор от малки услуги, всяка от които работи в свой собствен процес и комуникира с леки механизми“.

Архитектурите на миктоуслугите придобиват голяма популярност поради непрекъснато нарастващата сложност на поддръжката и мащабирането на монолитни приложения. Те са разработени под концепцията „направи едно нещо и го направи добре“, т.е. всяка микроуслуга трябва да бъде фокусирана върху една бизнес услуга, която може да бъде доставена и актуализирана като независима система.

Според AurelioVivas и JohnSanabria (Vivas, A., 2020), друг аспект, който трябва да се има предвид при внедряването на микроуслуги, е използването на контейнери. Това е

полезно, тъй като отделни услуги могат да бъдат хоствани в отделни контейнери. Тези контейнери загражда самите микроуслуги, включително всички необходими библиотеки и данни, което позволява на микроуслугите да бъдат самостоятелни.



Фиг. 5. а) монолитна архитектура; б) архитектура на микросервиз.

ИЗВОДИ

Проектирането и изграждането на системи за автоматично инсталиране и внедряване на софтуер е сложен и комплексен процес, който е с изключително голямо търсене, и значение за напредъка на технологиите в световен мащаб. Разработката на системи за доставянето на услуги с универсален достъп на потребителя, централизирано администриране на приложенията, делегирането на конкретни права към потребителия, стартирането на приложенията върху различни релани или виртуални платформи, събирането, съхраняването и обратката на различни типове данни, са водещите тенденции в съвременните информационни инфраструктури.

REFERENCES

Leymann, F., Fehling, C., Mietzner, R., Nowak, A., & Dustdar, S. (2011). *Moving applications to the cloud: an approach based on application model enrichment*. International Journal of Cooperative Information Systems, 20(03), 307-356.

Arcangeli, J. P., Boujbel, R., & Leriche, S. (2015). *Automatic deployment of distributed software systems: Definitions and state of the art*. Journal of Systems and Software, 103, 198-218.

Zhang, J., Ma, M., He, W., & Wang, P. (2020). *On-Demand Deployment for IoT Applications*. Journal of Systems Architecture, 101794.

Pressman, R. S. (2005). *Software engineering: a practitioner's approach*. Palgrave macmillan.

Butzin, B., Golatowski, F., & Timmermann, D. (2016). *Microservices approach for the Internet of things*. in 2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA) (pp. 1-6). IEEE.

Vivas, A., & Sanabria, J. (2020). a *Microservice Approach for a Cellular Automata Parallel Programming Environment*. Electronic Notes in Theoretical Computer Science, 349, 119-134.

Rees-Carter S., Drake M., E. Heidi, (2018). How to Install and Configure Ansible on Ubuntu 18.04.

Potdara A. M., Narayan D Gb, S. Kengondc , M. Moin Mullad. (2020). *Performance Evaluation of Docker Container and Virtual Machine*. Third International Conference on Computing and Network Communications, pages 1–1.