25 AVRIL 2020

ÉCOLE D'INGÉNIEURS INFORMATIQUE

PROSIT 13

QUENTIN DAVEAU
A4 EXIA CESI
Centre de La Rochelle

Table des matières

I) Introduction

A) Contexte

Mélanie veut installer un environnement pour développer un outil informatique, elle encontre des soucis de compatibilité entre les différents environnements, elle doit proposer une solution pour avoir un environnement uniforme et compatible entre les OS.

B) Contraintes

Les besoins définis ont été les suivants :

- Nada, Niet, que pouick, rin du tou

C) Problématiques

Les problématiques qui ont été levées sont les suivantes :

- Comment peut-on installer un environnement de développement fonctionnel ?
- Comment choisir des outils d'environnement adaptés à nos besoins ?
- Comment proposer un package viable dans le futur peu importe les MAJ et les OS sur lequel il tourne ?

D) Hypothèses

- En quoi consiste le SGS ?
- Quel est l'intérêt d'utiliser un Framework ?
- Tableau comparatif entre intelliJ, NetBeans et Eclipse
- Comment résoudre rapidement les problèmes de compatibilité avec l'OS
- Utiliser Docker ou une machine virtuelle permet d'éviter les soucis de compatibilité avec l'OS
- Comment s'organiser en projet ou en stage avec les collègues ?
- Quels outils utiliser pour faciliter la collaboration ?
- Avez-vous déjà installé des environnements de travail ?
- Si oui, avez-vous rencontré des difficultés ? Si oui, comment les avez-vous contournées ?

II) Mots clés

SGS : C'est le nom de l'entreprise de l'énoncé, rien de plus

Tools : Outils, wow ! Incroyable ! Mais qui aurait donc bien pu le deviner ?

Java EE : Le terme « Java EE » signifie Java Enterprise Edition, et était anciennement raccourci en « J2EE ». Il fait quant à lui référence à une extension de la plate-forme standard (Java SE). Autrement dit, la plate-forme Java EE est construite sur le langage Java et la plate-forme Java SE, et elle y ajoute un grand nombre de bibliothèques remplissant tout un tas de fonctionnalités que la plate-forme standard ne remplit pas d'origine.

Framework : Un désigne un ensemble cohérent de composants logiciels structurels, qui sert à créer les fondations ainsi que les grandes lignes de tout ou d'une partie d'un logiciel (architecture).

SDK : Un kit de développement logiciel, aussi appelé trousse de développement logiciel, est un ensemble d'outils logiciels destinés aux développeurs, facilitant le développement d'un logiciel sur une plateforme donnée (par exemple, iOS, Android, Linux, OS X, Microsoft Windows).

Payara server : Payara Server est un serveur d'applications open source dérivé du serveur d'application GlassFish en édition libre.

Installation d'environnement : L'installation d'un framework. Cela peut causer des contraintes en fonction de l'OS…

Guide d'installation : Semblable au guide pour aveugle, le guide d'installation aide les personnes à mobilité réduite à s'installer quelque part, sur une chaise, dans un lit…

I) Outils de développement

IDE :

Un IDE ou un Environnement de développement (Integrated Development Environment) est un logiciel qui rassemble des outils permettant de développer d'autres logiciels tels que des applications mobiles, des logiciels pour ordinateur ou consoles de jeux, des sites web, etc... ainsi que de réaliser des librairies ou des Framework, c'est-à-dire des morceaux de code qui pourront être sauvegardés et réutilisés dans d'autres programmes.

Les outils d'un IDE peuvent être :

- Un éditeur de code intelligent (Coloration, autocomplétions, mise en forme) ;
- Un simulateur (logiciel permettant de tester l'exécution de son logiciel) ;
- Un compilateur (qui va transformer le code source rédigé par le développeur en code binaire) ;
- Un débogueur (fonctionnalité d'aide à la correction de bugs).

Il existe de nombreux IDE, certains permettent de développer pour un système d'exploitation spécifique, d'autres sont polyvalents.

A) Java

Java SE/EE :

La plate-forme Java (the Java Platform en anglais, plateforme Java 2 anciennement) est un standard de facto de plate-forme logicielle, produit par Sun Microsystems, puis Oracle Corporation, permettant de développer et d'exécuter des programmes écrits en langage Java indépendants de tout processeur et de tout système d'exploitation, conformément à la technologie Java.

Toute plate-forme Java se compose principalement d'un moteur d'exécution (appelé une machine virtuelle Java, ou 'JVM') et d'un compilateur fourni avec un ensemble de bibliothèques standards dont il existe plusieurs implémentations pour divers matériels et systèmes d'exploitation, de façon que les programmes Java puissent s'exécuter de façon identique sur chacun d'entre eux.

Le terme Java Platform est avant tout une dénomination commerciale. Sa désignation comme Plate-forme plutôt que Framework se justifie cependant par l'intégration, avec les librairies logicielles constituant le framework Java, de la JVM qui permet d'exécuter les programmes Java sur différents OS.

Ainsi, chaque plate-forme Java ne se limite pas à une bibliothèque de composants mis à disposition, elle se caractérise de surcroit par sa faculté à exécuter des logiciels.

Plus de détails :

https://www.theserverside.com/definition/J2EE-Java-2-Platform-Enterprise-Edition

Les différents environnements Java (EE, SE…) :

https://docs.oracle.com/javaee/6/firstcup/doc/gkhoy.html

Les IDE Java, il y en a plein :

NetBeans :

NetBeans est l'environnement de développement libre et open source le plus puissant pour Java. Il s'agit de l'IDE officiel pour Java.

Il est très utilisé par les développeurs professionnels pour le développement d'application d'entreprise, web, mobile ou encore bureautique.

NetBeans est un IDE multi-plateforme. Il peut donc supporter Linux, Windows, Mac ainsi qu'Oracle Solaris.

Il est aussi un environnement de développement de bout en bout (end to end) qui facilite l'analyse, la conception, le codage, le profilage, le test, le débogage, la compilation, l'exécution et le déploiement d'applications.

Il s'intègre aussi parfaitement au développement avec les API de persistance java, JSP, spring, Struts, servlets, services Web et Framework Hibernate.

Ce qui différencie NetBeans des autres est son générateur d'interface graphique qui permet de créer des applications de bureau en utilisant Swing et NetBeans Platform.

En plus de Java, NetBeans permet la prise en charge native de divers langages tels que le C, le C++, le JavaScript, le XML, le Groovy, le PHP, le HTML ou d'autres (dont Python et Ruby) par l'ajout de greffons mais il reste le plus performant en termes de développement en java et des technologies associés.

IntelliJ Idea :

IntelliJ IDEA aussi appelé simplement Intellij est un environnement de développement complet les développeurs Java EE et tout ce qui est Java. IntelliJ a été créé par JetBrains.

La premier version d'Intellij est rendue publique en janvier 2001 et est, à ce moment-là, le seul IDE pour Java disponible disposant de possibilités de navigation avancée dans le code et de refactorisation de code.

IntelliJ est un IDE professionnel et se décline en deux versions :

- Une version gratuite (Free Community Edition) qui aura tendance à viser les développeurs seuls, à leur compte.
- Une version payante (Edition " Ultimate ") très avancée qui s'adresse cette fois ci aux développeurs travaillant dans des entreprises (Les licences coutent très chères).

La Free Community Edition aura de nombreuses fonctionnalités pour crées des applications Android ou encore des application JVM tandis que la version " Ultimate " disposera d'un ensemble plus moderne de fonctionnalité pour développer de manière productive des application Web et Java EE.

Elle apportera de nombreuses fonctionnalités supplémentaires tels que :

- Spring Java MVC, Spring Security, Spring Boot, Spring Integration et d'autres ;
- Support pour les Framework comme Node.js, Angular et React ;
- Support pour le développement des langages web comme Javascript, Typescript, Coffeescript etc... ;
- Prise en charge de Java EE, JSF, JAX-RS, CDI, JPA, etc... ;
- Contrôle de version avec Team foundation server, Perforce, Clearcase and Visual SourceSafe ;
- Support de Grails, GWT, Griffons et Vaadin ;
- Et le déploiement est supporté par presque tous les serveurs, y compris, TomCat, TomEE, GlassFish, JBoss, WildFly, Weblogic, WebSphere, Geronimo et Virgo.

En bref Intellij IDEA est idéale pour le développement en entreprise.

Eclipse :

Eclipse est un IDE que tous les développeurs JAVA connaissent forcement.

Eclipse a un écosystème propre avec une énorme communauté de développeurs, une documentation très fournie et beaucoup de plugins.

Les développeurs utilisent Eclipse pour développer des applications mobiles, de bureau, web et des systèmes embarqués.

Eclipse est principalement écrit en Java et est disponible gratuitement en open source sous une licence publique Eclipse.

Il est utilisable sur des plateformes comme Windows, Mac OS X et Linux.

Eclipse est un peu le papa des environnements de développement. Beaucoup d'IDE gratuits et commerciaux sont construits sur Eclipse comme MyEclipse, Orion ou encore RAD.

Il peut être utilisé pour la programmation dans de nombreux autres langages que JAVA et est considéré actuellement comme le meilleur environnement de développement disponible.

Il existe tout un tas d'autres IDE, mais tous les citer ici serait superflus, pour une liste plus exhaustive, voir :

https://www.supinfo.com/articles/single/6135-top-10-ide-developpeurs-java

B) .NET

.NET Framework est, comme son nom l'indique à moitié, à la fois une framework et un environnement runtime. Une framework est une collection de code tout fait et prêt à l'usage pour les développeurs. Il leur permet donc de réutiliser du code existant, non seulement pour gagner du temps, mais aussi pour permettre une certaine standardisation entre les applications. Le codeur peut alors se concentrer sur le code qui rend réellement son application unique. Mais ce n'est pas son seul attrait. Car il s'agit également d'un environnement runtime.

Un peu comme les iFrame en HTLM, cet environnement est indépendant du système. Il créée un cadre dans lequel les applications créées via .NET Framework peuvent se lancer indépendamment du système d'exploitation. Ce qui représente un intérêt évident : on peut lancer les applications codées avec .NET Framework sur n'importe quel système d'exploitation où .NET Framework peut être installé.

Plus d'infos :

https://docs.microsoft.com/en-us/dotnet/framework/get-started/overview

https://docs.microsoft.com/fr-fr/dotnet/standard/components

Des IDE .NET, y'en a aussi plein. Globalement, c'est Visual Studio le plus connus et le meilleur, cependant il existe de nombreuses alternatives, comme Eclipse avec le plugin aCute, ou Project Rider, la version JetBrains :

https://teckangaroo.com/best-ide-for-c-that-you-can-use-in-2020/#codeplugin

C) Android

De nos jours, Android est le système d'exploitation le plus utilisé dans le monde avec une part du marché qui atteint les 80 %, devant l'iOS d'Apple et Windows Phone de Microsoft (lol).

Donc, il n'est pas surprenant que de nombreux développeurs cherchent à créer de nouvelles applications fonctionnant sous ce système d'exploitation basé sur Linux, que ce soit dans le but de se faire plaisir ou pour gagner de l'argent.

Cependant, qu'il soit un amateur ou un expert dans le domaine, un développeur sait incontestablement que pour mener à bien son projet, il doit :

- Apprendre le développement mobile.
- Connaître les bases du langage Java ou en général la programmation orientée objet.
- Installer un bon nombre de softwares sur son ordinateur comme le kit complet SDK Android ou encore le JDK.
- Choisir le bon environnement de travail.

Afin de réduire les coûts et les délais de réalisation d'une nouvelle application, de nouveaux environnements de travail qui rassemblent divers outils, sont lancés chaque année pour aider les développeurs dans leur labeur.

Ainsi, les IDE, Integrated Development Environment, facilitent la compilation, l'écriture et le débogage du programme réalisé et permettent d'obtenir le fichier exécutable (.apk). Les plus connus sont Android studio ou Eclipse. Il existe aussi des environnements cross-platform (Android et IOS) comme Xamarin (extension de visual studio) et Titanium.

Plus de détails sur les IDE :

https://www.appstud.com/fr/guides/agence-mobile/app088/

https://www.developer.com/ws/android/development-tools/top-android-ides-for-developers.html

D) Parenthèse virtualisation

La virtualisation est une technologie qui vous permet de créer des services informatiques utiles à l'aide de ressources qui sont généralement liées au matériel. Elle vous permet d'exploiter toute la capacité d'une machine physique en la répartissant entre de nombreux utilisateurs ou environnements différents.

Pour plus d'infos sur la virtualisation, voir :

https://www.redhat.com/fr/topics/virtualization/what-is-virtualization

E) Parenthèse serveur d'application

Un serveur d'applications est un logiciel d'infrastructure offrant un contexte d'exécution pour des composants applicatifs. Le terme est apparu dans le domaine des applications web. Au sens strict les composants hébergés par le serveur d'applications ne sont pas de simples procédures ou scripts mais de réels composants logiciels conformes à un modèle de composants (EJB, COM, Fractal, etc.).

La structuration en couches des différents composants mis à disposition par le serveur d'application permet une prise en compte des besoins métier, des interactions avec les utilisateurs, des connexions avec les bases de données, etc.

Les serveurs d'applications sont des logiciels occupant la couche centrale dans une architecture multicouche, qu'elle soit classique trois tiers (postes clients, serveur d'applications, serveur de données) ou étendue N tiers lorsqu'elle intègre par exemple des serveurs d'acquisition (données de terrain, données de process, de back-office, etc.) ou des serveurs d'interface (gateways, systèmes coopérants externes, etc.).

Dans un sens plus large, un serveur d'applications peut être une machine servant à héberger des applications, soit pour permettre leur exécution depuis un poste client (mode client-serveur de données, généralement partage de fichiers et politiques de gestion des accès), soit pour déporter l'affichage sur le poste client (mode client-serveur d'affichage).

Utilisés, à l'origine, pour concevoir des pages web dynamiques, les serveurs d'application sont devenus les infrastructures des nouvelles applications bâties à l'aide de composants métier et intégrées aux systèmes d'information des entreprises.

Détails :

https://www.01net.com/actualites/comprendre-le-role-des-serveurs-dapplication-192775.html

Serveurs d'application Java et JavaEE :

https://blog.idrsolutions.com/2015/04/top-10-open-source-java-and-javaee-application-servers/

III) Outils de collaboration

A l'heure de la digitalisation et des nouvelles formes de travail, les outils de collaboration se multiplient. Du réseau social d'entreprise aux outils de gestion de projet, il en existe aujourd'hui de toutes les sortes, si bien qu'il convient d'en étudier les différentes fonctionnalités, l'ergonomie et leur coût pour choisir le bon.

On peut distinguer trois catégories d'outil :

- Les réseaux sociaux d'entreprise (RSE) qui sont souvent généralistes ;
- Les suites bureautiques en ligne qui intègrent pour certaines une dimension sociale et collaborative (Google Apps, Office 365, …) ;
- Les solutions collaboratives de ChatOps (Slack, Jamespot, TalkSpirit, Microsoft Teams, …) qui incluent des chatbots, c'est-à-dire des agents conversationnels (des robots) capables d'analyser et de répondre automatiquement à une question posée sur une messagerie instantanée.

Il existe de nombreux outils de collaboration, globalement dans deux catégories, ceux dédiées aux grandes entreprises (Asana, Zoho projects) et pour des projets de grande envergure (LiquidPlanner), et ceux dédiés aux petites entreprises (Slack, Trello, Teams…).

Certains outils de collaboration sont plus spécialisés que d'autres. Asana et Podio sont très polyvalents par exemple.

Pour récap :

- Polyvalence : Podio, Asana
- Management de ressources et de projets : LiquidPlanner, Zoho Projects
- Management des tâches et du workflow : Asana, Airtable
- Kanban : Asana (the best), LeanKit (pour Agile), Volerro (pour assets visuels), Wrike (plus généralisé)
- Communication d'équipe : Slack (the best), Glip (moins cher)

Plus d'outils et de détails :

https://www.pcmag.com/picks/the-best-online-collaboration-software

IV) Livrables

Les corbeilles sont juste des installations, donc il n'y a pas grand-chose à montrer ici.

V) Bilan

A) Validation des hypothèses

En revenant sur les hypothèses mises en place au début du prosit :

- **En quoi consiste le SGS ?**

C'est le nom de notre entreprise, non ?

- **Quel est l'intérêt d'utiliser un Framework ?**

Normaliser les fonctions utilisées, ne pas avoir à refaire toutes les fonctions basiques

- **Tableau comparatif entre intelliJ, NetBeans et Eclipse**

Voir https://stackshare.io/stackups/eclipse-vs-intellij-idea-vs-netbeans#more

- **Comment résoudre rapidement les problèmes de compatibilité avec l'OS**

Travailler sur un environnement normalisé

- **Utiliser Docker ou une machine virtuelle permet d'éviter les soucis de compatibilité avec l'OS**

Oui, surtout une machine virtuelle

- **Comment s'organiser en projet ou en stage avec les collègues ?**

Utiliser des outils de collaboration plus ou moins avancés en fonction des besoins et du contexte

- **Quels outils utiliser pour faciliter la collaboration ?**

Il y en a plein, beaucoup d'outils basiques peuvent être utilisés peu importe le projet (Trello, Miro…)

- **Avez-vous déjà installé des environnements de travail ?**

Oui

- **Si oui, avez-vous rencontré des difficultés ? Si oui, comment les avez-vous contournées ?**

Quelques problèmes de version de logiciel créant des incompatibilités avec le reste de l'équipe. Arrangés en normalisant la version utilisée.

B) Réponse aux problématiques

Les problématiques étaient :

- **Comment peut-on installer un environnement de développement fonctionnel ?**
- **Comment choisir des outils d'environnement adaptés à nos besoins ?**
- **Comment proposer un package viable dans le futur peu importe les MAJ et les OS sur lequel il tourne ?**

Il n'y avait pas de contraintes.

De nombreux outils peuvent entrer en jeu dans la mise en place d'un environnement de développement, ils consistent généralement en un framework (relatif au langage, .NET ou Java EE/SE par exemple), un IDE (Visual studio, Eclipse…), un serveur d'application (Payara…). Le choix de l'environnement est très spécifique, relatif aux besoins, au langage… Mettre en place l'environnement sur une machine virtuelle ou utiliser un serveur d'environnement permettent de normaliser l'environnement et de s'assurer que celui-ci sera compatible peu importe l'OS.

VI) Sources

Voir liens tout au long du prosit. Eh ouai, c'est de belles sources. Nan mais qui regarde les sources de toute façon ? Genre c'est tout à la fin du prosit, et hors-contexte. Ce n'est pas adapté pour un document d'apprentissage.