# تحديد المتطلبات

## نظرة عامة

بعدما قضينا بعض الوقت للتعرّف على النطاق والشكل العام والخصائص الأساسية للنظام ، سنحتاج الآن للدخول إلى مرحلة أكثر تفصيلاً عن طريق تحديد متطلبات النظام ، مالذي نريد أن يقوم به النظام.

في هذه المرحلة سنظل مهتمين بمالذي سنقوم بعمله وليس بكيفية عمله. وفي هذا الموضوع سنستعرض بعض الأفكار والأساليب الهامة التي تستخدم لتحديد متطلبات النظام.

## تحليل المتطلبات - Requirements analysis

تحليل المتطبات هي عملية إنتاج وصف مفصّل للنظام يحتوي على الخواص التالية:

- كامل (complete)
- واضح (unambiguous)
- تنسيقه وطريقة عرض المعلومات فيه ثابتة (consistent)
- دقيق (precise)
- قابل للإثبات (verifiable)
- منعزل عن عملية التنفيذ والتطبيق (implementation independent)
- سهل القراءة (readable)
- قابل للتحرير (modifiable)
- مرتب ومنسّق بشكل جيّد لتسهيل مراجعته والرجوع إليه (well-organized for reference and review)

## لماذا نحتاج إلى تحديد متطلبات البرمجيات؟

يقول بروكس (Brooks 1987): "أصعب جزئية في بناء نظام برمجي هي تحديد ماذا ستبني بشكل دقيق. لا توجد جزئية أخرى بمثل صعوبة تحديد المتطلبات التقنية بشكل دقيق ، ومن ضمنها تحديد واجهات الاستخدام مع الناس ومع الأجهزة ومع أنظمة برمجية أخرى. لا توجد جزئية أخرى ستتسبب في فشل النظام إذا تم عملها بشكل خاطئ. ولا توجد جزئية أخرى يصعب تصحيحها في المستقبل مثل جزئية تحديد المتطلبات."

في موضوع "كيف تحلل المشكلة" تحدثنا بشكل مختصر عن حاجات (needs) المستخدم وأصحاب المصلحة وخصائص ( features) النظام و حالات الاستخدام (use cases). وكنا نستخدمها بغرض وصف النظام بشكل عام.
الحاجات تصف القضايا المتعلقة بنطاق المشكلة "problem domain" ، وهي تعطينا تصور مبدئي عن مالذي يجب أن يقوم به النظام ليشبع هذه الحاجات.
الخصائص تقع بين حاجات المستخدم الحقيقية وبين تفاصيل كيف يقوم النظام بتحقيق الحاجات ، وهي أول خطوة للوصول إلى حل المشكلة.
الحاجات والخصائص يفتقدون إلى تفاصيل تقنية ولكنهم يعطون تصوراً بلغة بسيطة وسهلة وغير تقنية عن مالذي سيقوم به النظام حتى يلبي هذه الحاجات.
حالات الاستخدام هي وسيلة لوصف كيف سيتفاعل ويتواصل المستخدم مع النظام لتحقيق أحد خصائص النظام.
عندما نجمع الثلاثة أشياء معاً سنكون قد عملنا "ورقة الرؤية" (Vision document) للنظام بحسب تسمية Dean Leffingwell. ويقول أيضاً بأن هذا الملف أو التقرير لن يحتوي على تفاصيل متعمّقة وبالتالي لا يصلح أن نقوم بتسليمه إلى فريق التطوير حتى يقوموا ببناء النظام. ولذلك سنحتاج إلى وصفاً متعمقاً نعرّف فيه مالذي يجب على النظام عمله ، وهذا يتم عن طريق تحديد متطلبات البرمجيات.

## مالمقصود بالمتطلبات؟

يقول Dean Leffingwell : "متطلبات البرنامج هي قدرة في البرنامج يحتاجها المستخدم حتى يستطيع حل مشكلة تساعده في تحقيق هدفه ، أو قدرة في البرنامج يجب توفيرها من قبل النظام لمطابقة العقد أو المواصفات القياسية أو أي ملفات رسمية أخرى."

ويقترح دين بأن نستعمل هذا التعريف كدلالة على أنه يجب علينا تطوير متطلبات النظام الدقيقة عن طريق تصفية أو إزالة بعض الخصائص المقترحة للنظام. كل متطلّب سيخدم أحد الخصائص ، وأغلب الخصائص ستقود إلى عدة متطلبات متشابهة ومترابطة. نحن هنا لانطلب كتابة متطلبات جديدة لاترتبط بأحد الخصائص التي تم تعريفها ، ولكننا عندما نكتب المتطلبات من المحتمل أن نحتاج إلى إعادة كتابة الخاصية أو حتى إضافة خاصية جديدة لم نكن نفكر بها من قبل. فمهما حاولنا أن نفكِّر ونكتب كل الخصائص في البداية ، بعضها لن يظهر لنا إلا في النهاية.

والغرض الأساسي من مرحلة تحديد المتطلبات الموجودة في دائرة حياة البرمجيات هو أن نحدد بدقة ماذا نريد أن يفعل النظام. ويجب أن نتفادى عملية التفكير في كيفية قيامه بها لأنها تتبع لمرحلة أخرى لاحقة. محاولة فصل "ماذا نريد" عن "كيف سنقوم بـ.." هي عملية ليست بالسهلة في الواقع كما تبدو نظرياً ، ففي الأولى (وهي المتعلقة بتحديد المتطلبات) سنركز على فهم وتحديد المشكلة ، وفي الأخرى سنركز على فهم وتحديد الحلول المناسبة لهذه المشكلة.

## أصناف المتطلبات

توجد ثلاثة تصنيفات يتفق عليها جميع الكتّاب وسنضيف عليها تصنيفاً رابعاً أصبح مهمّا بسبب ظهور الأنظمة التفاعلية الحديثة وهو "متطلبات واجهة الاستخدام". التفرقة بين التصنيفات الأربعة سيكون صعباً سواء في تعريفها أو في استخدامها ، ولكننا سنتحدث عنها بشكل موجز حتى تعطينا إطار بسيط نستطيع أن نتحدث عن ونفهم الأنظمة من خلاله.

### 1- المتطلبات الوظيفية - Functional requirements:

الهدف من هذه المتطلبات هو أن تقوم بتعريف النواحي الوظيفية في النظام. فهي تصف ماهي مدخلات النظام وماهي المخرجات ، وكيف نقوم بتحويل المدخلات إلى المخرجات التي نرغب بها.

كمثال بسيط على المتطلبات الوظيفين : "عندما نتحدث عن نظام يقدم خدمات حجز طيران عن طريق الانترنت ، المستخدم يجب أن يكون قادراً على البحث عن الرحلات المتاحة بحسب التاريخ والوقت ، ويجب أن يعرف الأسعار لمختلف الرحلات ، ويجب أن يكون قادراً على تحديد تفضيلاته من ناحية هل يفضل أن تكون الرحلات مباشرة أو لا مانع لديه إذا لم تكن مباشرة." ولنظام معلومات خاص بأحد المكتبات: "يجب أن يستطيع المستخدم أن يبحث عن الكتب بواسطة اسم الكتاب أو الكاتب أو الناشر."

### 2- المتطلبات الغير وظيفية - Non-functional requirements:

هذه المتطلبات من المفترض أن تصف الخصائص الإضافية للنظام مثل متطلبات الأداء أو الطاقة الإنتاجية أو قابلية الاستخدام للنظام أو أمان النظام. فهي تعتبر متطلبات للجودة الإجمالية للنظام ، ومن الممكن أن تضع أو تضيف هذه المتطلبات بعض القيود على المنتج الذي يتم تطويره أو على طريقة تطويره.

وكمثال عليها من الممكن أن نضيف متطلب أمان للنظام ، "جميع البيانات يجب أن تكون محمية من أي دخول غير مصرح به". ومن الممكن إذا قمنا بتفصيل هذا المتطلب أن نصل إلى متطلبات وظيفية متفرّعة منه.

### 3- قيود التصميم - Design constraints:

بدلاً من تعريف مالذي سيعمله النظام ، هذا التصنيف يختص بتحديد بعض القيود على تصميم النظام والتي يجب أن نلبيها حتى يتوافق النظام مع المواصفات التقنية المطلوبة أو مع مواصفات العقد.

كمثال عليه من الممكن أن نقول "يجب أن تتم كتابة البرنامج بلغة الجافاسكربت ويجب أن يعمل بشكل سليم مع متصفح مايكروسوفت انترنت إكسبلورر". أو من الممكن أن نقول "يجب أن يستطيع فاقدي البصر من استخدام النظام بدون مشاكل". ومن الممكن أن يتوجب علينا أن نصمم النظام بحيث يستطيع أن يتفاعل مع أنظمة أخرى.

### 4- متطلبات واجهة الاستخدام - User interface requirements:

من الممكن أن يكون للنظام عدة طبقات وأنواع من المستخدمين من حيث نوعية استخدامهم للنظام (متعمق أم بسيط) أو من حيث معرفتهم وخبرتهم في التعامل مع مثل هذا النظام ، ولذلك يجب علينا تحديد متطلبات من شأنها تسهيل استخدام النظام لجميع الفئات.

## تسلسل المتطلبات الوظيفية

في أغلب الحالات ستكون المتطلبات مرتبطة مع بعضها البعض (خاصة المتطلبات الوظيفية) ، ويمكننا أن نقوم بتنظيمها وتلخيصها على عدة مراحل بحيث تكوّن لنا شكل هرمي أو شجري. فالمتطلبات المعقّدة يمكن تقسيمها بشكل أفقي حتى يصبح لدينا عدة متطلبات بسيطة (أفضل من متطلب واحد معقّد) ، وفي كل من هذه المتطلبات الأفقية يمكننا تفصيل كل متطلب بشكل عمودي بشكل أكبر.

## كيف نقوم بتوثيق المتطلبات؟

وثيقة المتطلبات هي وثيقة رسمية لمتطلبات النظام موجهة للعملاء و المستخدمين النهائيين و مطوري البرمجيات. اسم هذه الوثيقة من الممكن أن يختلف من منظمة إلى أخرى ، ولكن غرضها وهدفها ثابت مهما تعدد اسمها ، وهي تعتبر النتيجة النهائية المتفق عليها في مرحلة تحليل المتطلبات. وسيبنى على أساسها جميع مراحل تطوير النظام اللاحقة. وهي تعتبر كمصدر للمعلومات لكل من يهتم بالنظام لأنها تصف الاتفاق العام على ماهو النظام الذي سيتم بناؤه.

تنسيق هذه الوثيقة يختلف من منظمة لأخرى ويختلف من نظام لآخر بحسب تعقيده وحجمه.

## ترتيب المتطلبات بحسب أهميتها - Prioritising requirements

بعدما نقوم بتحليل النظام وتحديد وتوثيق جميع المتطلبات المحتملة من المفضل أن نتوقف قليلاً ونفكّر بواقعية أكثر. فمن الممكن أن لايتوفر لدينا وقت كاف لتلبية جميع المتطلبات ، أو من الممكن أن لايتوفر المال اللازم أو غيرها من الظروف الأخرى. في هذه الحالة يجب علينا أن نقوّم بترتيب المتطلبات بحسب أهميتها حتى نستطيع أن نؤجل أو نلغي المتطلبات الغير مهمة بحسب الظروف.

Dai Clegg قام بوضع استراتيجية بسيطة تساعد على ترتيب الأولويات. هذه الاستراتيجية تتلخّص في اسم العاصمة الروسية **MoSCoW** , وتفصيلها كالتالي:

- **M**ust have: وهي تنطبق على جميع المتطلبات الأساسية في النظام والتي من دونها لن يعمل النظام بشكل صحيح.
- **S**hould have: وهي تنطبق على المتطلبات التي كانت ستعتبر من ضمن المتطلبات الأساسية في حالة عدم وجود ضغوط زمنية أو مالية. بدون هذه المتطلبات سيعمل النظام كما هو متوقع منه بدون أية مشاكل.
- **C**ould have: وهي تنطبق على المتطلبات التي ستكون مفيدة ولكن سيتم إضافتها فقط في حالة سماح القيود الزمنية والمالية بذلك.
- **W**on't have: تنطبق على المتطلبات التي تعتبر مفيدة ولكنه سيتم تأجيلها إلى تطويرات أخرى للنظام.

وبإمكان فرق العمل أن تعتمد نظام التصويت على المتطلبات حتى تضمن وجود المتطلبات الأساسية في نهاية مرحلة تحليل المتطلبات للمشروع.

## الخلاصة

هذا الموضوع لا يهدف إلى تقديم تغطيةٍ شاملة لعملية تحديد المتطلبات ، وإنما يهدف إلى إعطاء فكرة وتصوّر شامل لهذا الموضوع. لأن هذا الموضوع واسع جداً ويتم تدريسه في مقررات دراسية متخصصة كاملة.