مقدمة:

في كل نظام برمجي، لا ينتج التصميم النهائي لبنية النظام مباشرةً، وذلك مهما كان دفتر الشروط الوظيفية، إن وجد، دقيقاً ومفهوماً. بل يتطور هذا التصميم عبر مراحل متلاحقة، ويتبلور حتى وصوله إلى التصميم الأمثل. طبعاً يتم ذلك عن طريق التوصيف الدقيق، والتحليل الشامل لكل أبعاد النظام، والفهم العميق لوظائفه ولآفاقه.

تصميم بنية النظام:

انطلاقاً من توصيف النظام (توصيف المعلومات، وتوصيف الوظائف)، ومن نتائج تحليل المتطلبات التي وضحت مفهوم التجربة أو الجلسة، تطورت بنية النظام التصميمية، ومرت عبر مراحل عديدة، وذلك مع تقدم فهم النظام وتبلور الرؤيا النهائية لآفاقه، إن كان على مستوى التعديل أو على مستوى التطوير والتوسيع.

يمكن جمع هذه المراحل في مرحلتين أساسيتين، يميزان فعلاً تقدم تصميم النظام. وهما:

- المرحلة الأولى: وفيها كان النظام ككل هو عبارة عن جلسة واحدة فقط. في هذه المرحلة، من أجل كل جهاز تم تعريف بنى ثابتة Structs لبارمترات الدخل والخرج، ومجموعة من البنى الأخرى لتوصيف الجهاز، بحيث يجري ملء هذه البنى أثناء البرمجة.

إن طريقة العمل هذه أدت إلى ضخامة حجم العمل البرمجي الروتيني، بالإضافة إلى بطئ تنفيذ البرنامج العام، وزيادة حجمه، بحيث أصبح من الصعب كشف الأخطاء.

ومن جهة أخرى، أصبح من المستحيل إكمال النظام حينما نتطرق لمفهوم الجلسة.

- المرحلة الثانية: تم فيها مواجهة مسألة تحقيق مفهوم الجلسة، وهذا بدوره استدرج ضرورة مواجهة مسألة إضافة جهاز جديد إلى النظام من قبل الزبون وليس من قبل الجهة المنفذة للنظام.

لحل المسألة الأولى مع مراعاة قيود المسألة الثانية، جرى العمل كالآتي:

عرفنا صف مجرد TDevice يكون بمثابة أب لكل الأجهزة، ويتضمن:

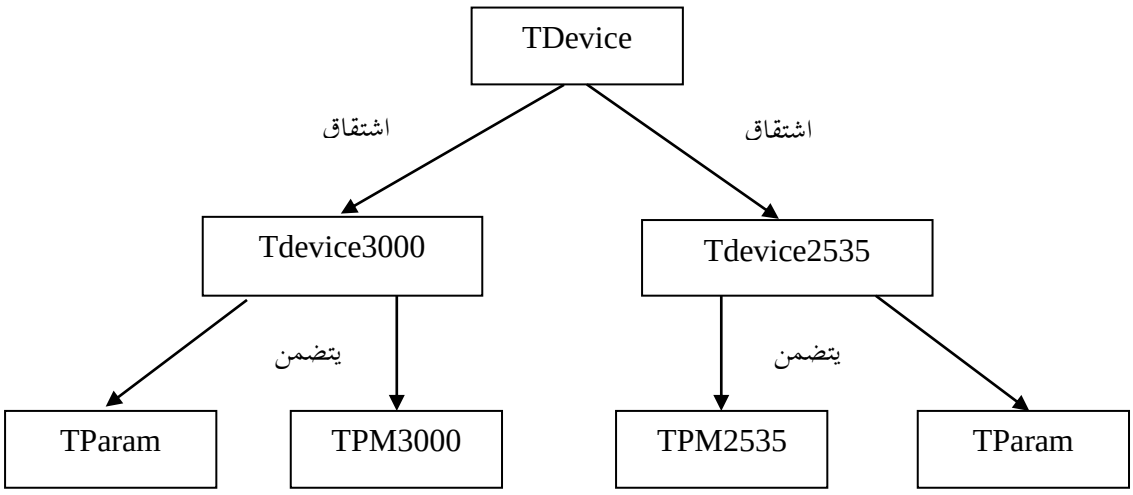
- إجراء لتهيئة الجهاز، أي وصله بشبكة الأجهزة المشاركة بالجلسة وإسناد العنوان المناسب له.
- إجراء لتهيئة إعدادات الجهاز.
- إجراء لحفظ وتحميل قيم بارامترات الجهاز.
- إجراء لتنفيذ عمليات القياس والتحصيل.

والآن من أجل تعريف جهاز جديد نقوم بما يلي:

- ✓ نشتق صف ابن من هذا الصف الأب، ونعرف الإجراءات السابقة بالقيم الخاصة بهذا الجهاز.
- ✓ نعرف صف يتضمن توابع قيادة هذا الجهاز Driver.
- ✓ نعرف صف يتضمن كل بارامترات هذا الجهاز، مع كل القيم الممكنة لكل بارامتر.

من أجل تقليص حجم البرنامج، قمنا بحفظ بارامترات كل جهاز في ملفات نصية، ليتم تحميلها أثناء تحميل الجهاز لحظة اختياره للمشاركة في جلسة ما.

فمثلاً بعد إضافة الجهازين PM3000, PM2535 يكون لدينا البنية التالية:



يوضح الشكل السابق أن بنى تخزين بارامترات ومحددات الجهاز، بالإضافة إلى توابع قيادة هذا الجهاز، أي Driver، كلها مضمّنة في هذا الصف الابن.

إن مواقع حفظ الملفات النصية ولواحقها محددة في هذه البنية، ويجب عدم الإخلال بها، كيلا ينتج أي خطأ أثناء استثمار هذا النظام. فمن أجل كل جلسة عمل ينشأ لدينا مجلد خاص بها، يحوي مجلد لكل جهاز مشارك بهذه الجلسة، والذي يحوي بدوره مجلد يضم ملفات الإعدادات، ومجلد يضم ملفات البارامترات، ومجلد يضم ملفات القياسات المحصلة ضمن هذه الجلسة من قبل هذا الجهاز.

مع هذه البنية أصبحنا قادرين على تعريف جلسة عمل تحوي مجموعة من الأجهزة في شعاع تخزين ديناميكي.

من أهم الفوائد العملية لهذه البنية المقترحة هي تسهيل بناء واجهة إدخال ديناميكية، تقيد المستثمر بإدخال قيم محددة وصحيحة، وذلك لأنه يقوم بالاختيار فقط وليس بالكتابة. من الواضح أن هذا يقلل عمليات التحقق من صحة المعلومات المدخلة.

ضمن بنية الجلسة، اضطرينا لتعريف واجهة عمليات خاصة بكل جهاز، وهذا يضعف ديناميكية البرنامج الكلي ،لأننا نضطر لبرمجة هذه الواجهة عند تعريف أي جهاز جديد، مع التقيد بالمكان المناسب للإضافة، وإعادة عمليات التنقيح والربط. أي عملية إضافة جهاز جديد للنظام مقترنة فقط بالجهة المنفذة للنظام، لأنها الجهة الوحيدة الملمّة بالبنية التفصيلية للنظام.

كما أن عملية الإضافة هذه تزيد من حجم البرنامج التنفيذي، بالتالي يتوجب علينا عندها إعادة إصدار نسخة جديدة من البرنامج تتيح استثمار الجهاز الجديد.

طبعاً هذه العملية مكلفة زمنياً بالنسبة للمستثمر، لأنه يتوجب عليه انتظار صدور النسخة المطورة التي تحوي الجهاز الجديد من قبل الجهة المنفذة. كما أنها عملية تتطلب حجم عمل كبير من قبل المطور، الذي يجب أن تظل بنية النظام واضحة لديه مع مرور الزمن.

من جهة أخرى، في حال استلزمت المتطلبات إجراء أي تطوير، كتطوير طرق معالجة القياسات، أو كتطوير بنية جهاز أو مجموعة أجهزة معينة، يكون الحل الوحيد لتحقيق هكذا تطوير هو إصدار برنامج جديد يواكب هذه التعديلات. لا شك أن هذا العمل يتطلب من الجهد والزمن الشيء الكثير.

نقطة أخيرة، إن الملفات النصية هي بشكل دائم عرضة للتعديل الخاطئ أو الحذف، مما يسبب أخطاءً أثناء التنفيذ يصعب التنبؤ بسببها، وتصبح عملية إعادة الملفات إلى وضعها السليم عمليةً توجب الحذر والدراية التامة بمحتويات هذه الملفات.

لحل جميع المشاكل السابقة، بدأ التفكير يتجه نحو بنية تلبي كلاً من المتطلبات التالية:

- إمكانية إدارة عدة جلسات:

ونعني القيام بالعمليات التالية:

- تعريف جلسة عمل جديدة، وتعريف الأجهزة المشاركة فيها، وتحديد بارامترات إعداد كل جهاز.
- إجراء عمليات القياس وتخزين النتائج في قاعدة معطيات خاصة بالجلسة للقيام بالعمليات المناسبة عليها (كرسم المنحنيات البيانية، إنشاء جداول إحصائية، ...).
- حفظ موصفات الجلسة.
- تحميل جلسة مخزنة.

نسعى نحو بنية تخزين أصغرية ومنطقية، بحيث تزيل كل قيود التخزين المفروضة في البنية السابقة.

يمكن أن تكون هذه البنية على الشكل التالي: يتضمن مجلد التطبيق الأساسي ثلاثة مجلدات، مجلد مكتبات الأجهزة (يتم تمييز كل مكتبة باسم يوافق الجهاز المعني)، مجلد الجلسات (يتم تمييز كل جلسة عن طريق الهدف منها أو تاريخها)، مجلد النتائج (يقوم البرنامج بتسمية كل ملف فيه بشكل آلي).

- إمكانية إضافة جهاز جديد إلى النظام:

بحيث تتم هذه العملية بزمن وحجم عمل أصغريين، وبدون أخطاء (نفترض هنا أن برنامج قيادة هذا الجهاز الجديد وبارامتراته ومحدداته متوفرة وصالحة)، ودون الدخول في العمل البرمجي. أي بمعنى أن تعريف جهاز جديد يتم خارج البرنامج الأساسي ذي الحجم الثابت (لدينا إصدار وحيد). طبعاً في المحصلة يجب أن يملك الجهاز الجديد كل إمكانيات الأجهزة المتوفرة في النظام سابقاً.

بشكل مختصر، يجب أن يقدم النظام قالباً كاملاً يتيح للمستثمر إضافة جهاز جديد بشكل فعال ويسير، بعيداً عن تعقيدات العمل البرمجي وقيوده.