Doctor's appointment

“Doctor's appointment” was specified as part of the UbiCompForAll project in order to identify potential users of the technology that will be developed in the project and to generate design ideas for the services these users might compose.

Summary

Ove creates a service to help his ageing mother get to her doctor's appointments. It keeps track of the appointments, reminding her when its time to go, it helps her find her way to the doctor's office, and it lets him keep track of her progress and alerts him of problems.

Problem description

Oda is 75 years old, and starting to get forgetful. She is finding it harder to manage on her own as she gets older and starts to forget things. This is something that will happen to virtually all people who live to an old age while physically healthy enough to manage on their own. She has trouble remembering her doctor's appointments and keeping track of the bus schedule.

Her son Ove is concerned for his mother but also very busy with his work. He wants to assist her, but he also wants her to manage on her own as much as possible. For Oda to keep her cognitive abilities and independence as long as possible she should exercise those cognitive abilities. If she is provided with the right assistance she can keep track of her appointments and feel safe on her own, helping her feel confident and independent, which will in turn help her stay mentally fit longer.

Main actor (s)

Oda (75) is starting to get forgetful, but she is still able to get around on her own. She has a mobile phone, and can call and receive messages with it, but doesn't send messages.

Ove (47) is Oda's son. He wants to provide assistance for his mother, but he is also very busy with his work. He is accustomed to using computers in his daily life, but in no way an IT professional.

Activity scenario

Ove connects to the service he has created for his mother from his own computer and enters the time and date of her next doctor's appointment. The service has access to the bus schedule, and calculates what bus Oda should use to get to her appointment in time and when she should leave the house in order to reach the bus on time.

A computer in Oda's home is running the service, using her TV for output. It alerts her of the appointment with a message on the screen at specific times: one day in advance, one hour before she must leave, and when it is time for her to leave the house. Her cellphone is also running the service, showing the same alerts. She must take it with her, and it alerts her of this with a sound and message when it's time to leave.

The service keeps track of Oda's position, and has a route with schedule specifying where she should be at what time relative to the bus and appointment times. The front door of her house is equipped with a sensor, so the service can register her leaving the house. Her position is tracked using her phone. Her phone shows a short message describing her current task or destination. On the way to the bus stop it shows what bus to take and at what time it is scheduled to leave her stop.

Messages are sent to Ove's phone and email to alert him of an upcoming appointment, and when his mother leaves the house, so he can be prepared to contact her if a problem occurs. And he is alerted if Oda deviates from the schedule, such as staying at home too long or not reaching a point on the route within a time limit. All events registered by the system, such as messages sent and points on the route reached, are logged. Ove can view the log from his computer and phone, if he wants to check on his mother's progress or on how well the service is working.

So Oda finds her bus stop, waits for the bus and rides it to the city centre. Her phone tells her at what stop she should get off. It keeps track of her position, and alerts her when her stop is coming up, all the while registering the events for Ove to check on. Leaving the bus and walking to her final destination, she meets an old friend. They start to talk, and Oda forgets the time. After a period with no change in her position, the system alerts Ove. He calls her phone to ask her why she is delayed, and reminds her to walk on soon to reach the doctor on time. Continuing, she reaches her doctor. The system will assist her in a similar way when returning home, without the time of an appointment to worry about.

Composition scenario

The need for the service arises because Oda forgot her last appointment with her doctor. She finds it difficult to remember the details about what bus to take and when to take it, but she is still able to get around on her own. Her son Ove wants to compose a service to help her while letting her manage on her own as much as possible. He wants to keep track of her progress when going to the doctor, and he wants to be able to help her even when he's at work. He wants to connect a calendar storing appointments with alerts and tracking of Oda's position, and from what he has heard about the composition system he believes it to be possible.

He composes the service on behalf of himself and his mother; she is the main user of the service but he also has roles in its use. He composes the service on his home computer. He has installed the composition software and has access to components available in an online database. He has high ambitions for his service. He wants it to provide all the support Oda needs with minimal user interaction, so that he doesn't need to worry about anything unless he gets an alert. He wants it to be accessible from different terminals. And he wants it to be general and expandable, so he can build on it in the future, adapting it to other appointments and destinations. But he has no previous experience with the composition system, so he must work in iterations. He will start simple, creating a usable service with some basic functionality, and add to it later.

As a start he connects a calendar storing appointments to the sending of alerts to Oda's phone and his own email. He also includes a logger to keep track of what the system does. The calendar and log viewer have their own online interfaces and can be used independently of the composition, so Ove doesn't need to consider user interfaces for his composition. This first version of the service was easy to put together and provides some basic functionality, but it requires appointment alerts to be entered in the calendar for all scheduling.

For the second version Ove wants to add more scheduled tasks and to have a common schedule to run for each appointment. The composition system lets him create a schedule, setting up different tasks to run at different times relative to the time provided by the calendar. Some of these tasks are simply the sending of an SMS or email. But Ove finds a building block providing a bus departure time based on when a person needs to be at a destination, and connects the output to the sending of a message, to inform Oda of what bus to use. And he finds a building block providing Oda's position and schedules it to be checked shortly before her appointment.

Here he adds a condition to the composition, having it send him a message only if she is at home at this time.

Now Ove has a fairly useful service running, and he has become more confident with the composition system. For the next version he turns his attention to user interfaces. Rather than using the online client of the calendar enabler, he finds a building block for entering appointments that comes with a user interface specification. He modifies this to only include the time of the appointment, with everything else “hard-coded” in the composition, so he minimizes the task of entering appointments and the possibility of errors. Rather than using SMS for messages to Oda he finds a building block for receiving and showing messages. Messages can include sound in addition to text. It can be instantiated on various devices, and Ove deploys it on Oda's phone, where it will run in the background and show messages received. He also gets a small computer for Oda, to be connected to her TV to show output from the services he creates. He adds new messages to the schedule. The scheduled tasks are otherwise the same, except that Ove adds some computations to his composition, finding the time Oda should leave the house based on the bus departure time and making some of the schedule depend on this time rather than just the appointment time.

For the final (so far) version of the service, Ove adds full tracking of Oda's position. He sets up a route, consisting of positions and time offsets for when they should be reached. The list of just the positions is fed to a tracker component. This sends out an event each time Oda's phone reaches a position in the list. It is connected to an event manager component which can register events from various sources. Given a list of events, each with a time and allowable delay, it sends out an alert to Ove if an event is not registered on time. The route is used as a list of events, so Ove is alerted if the route is not followed. Ove also adds a sensor to Oda's front door, with a component providing open and close events to the composition so that the leaving of the house can be used to trigger a message to Ove as well as an event in the event manager schedule.

Now Ove has reached his goals for his composition, and the service can aid Oda in all her future doctor's appointments. But it will probably be modified later. Oda's needs will evolve as she gets older and her cognitive functions get weaker; her need for guidance will increase. The service can be modified when Oda's needs change, or when new useful components and services become available. For instance, if the bus schedule later includes dynamic updates for delays, the composed service also needs to respond to changes in schedule. Incorporating more components within the same system, other tasks can be managed by the same logic, and all the tasks can be coordinated with each other. This way the system can be expanded into a full day planner that keeps track of all Oda's tasks, listing them on her screen and giving her notifications.

Alternative stories

This scenario has much room for customization. Ove composes and refines the service to fit the needs of his mother, and there are many possibilities:

- The service could ask for acknowledgements from Oda when she is prompted about an upcoming appointment, so Ove would get a confirmation that she received the message. Interacting with the service could be done with her cellphone, or with a remote control or touch-sensitive screen when using her TV. It's probably best to keep user input from Oda to a minimum, as she probably doesn't like to press buttons too much.

- A weather forecast enabler could be included, so the message for Oda to get ready for her appointment could include a suggestion for what clothes to put on. So far her cognitive abilities are mostly intact so she can select sensible clothes herself, but this could become more useful as she gets older.

- The door sensor could include a camera. Ove can get a picture of his mother leaving the house, giving him visual confirmation and letting him check if she is properly dressed.

- To make it easier for Oda and Ove to call each other, the cellphone clients of the service could include a function for calling the other, so it can be done from a menu or with the push of a single button.

- The service could make use of a map/navigation system on Oda's phone. It could show her current position and provide instructions on where to go based on the route to her doctor, either by voice or plotted on the map. The map could be engaged through an option on her phone, or automatically when the system detects she is having trouble following her route. Oda might find it too difficult to use a navigation system. As long as her cognitive abilities are good she can manage well without it. As she gets older she may find it harder to navigate on her own, but then it will also be much harder for her to use her phone for navigating. However, if she manages to learn its use now, she might be able to get some aid from it later.

- Oda might need to use several buses to get to her doctor, changing buses in the city centre. This would not require any new components, just more steps for Ove to set up as part of the route.

- The bus schedule could be dynamically updated with delays from the bus company, so if a delay is registered the system can take this into account, possibly changing what time Oda should leave the house.

- If the bus registers passengers embarking and disembarking, and these events can be made available, they can be included as events to track Oda's progress.

- Automatic ordering of a taxi would be a useful component in case there is a problem with the bus. Let’s say Oda reaches the bus stop, and waits for her bus to arrive. No delay is known to the system, but no bus arrives. A certain period of time after the bus should have arrived, Ove is notified and contacts Oda to ask why she's still at the bus stop. As there's no bus, he could decide to call a taxi for her. This could be set up in advance so it can be done with the press of a button. He might not be available to contact her at this time, so the system could be set up to call a taxi automatically if a delay is detected which prevents Oda from reaching her appointment in the normal way. Her name, phone number and other identification would be sent to the taxi central, along with her position.

- The medical centre where Oda is going to see her doctor could also be involved in the use of the service. A client could be installed on one of their computers. Then the staff could register her appointments without Ove's involvement and keep track of Oda's progress when she has her appointment. But the medical centre would probably want to keep their involvement to a minimum.

Properties

This service composition is quite complex in the end, with scheduling and tracking being the main tasks. The scenario demonstrates how a user can build such a service in iterations, starting very simple and adding and changing the composition as he gains experience. It seems to be possible for an end user to compose such a service.

Each iteration of the composition added new properties. In the first version we had a simple sequence of events, with the first component triggering the running of the others. In the second version we added scheduling, to trigger components at different times. Scheduling is a feature which has general interest as a composition mechanism. We also added a condition with branching.

The third version was concerned with user interfaces. Some components will come with a user interface. The composer should be able to configure it by selecting to show only some of the elements of the interface, and possibly reorder the shown elements. The most flexible approach to user interfaces would be to have the components include an abstract specification, saying what input and output elements the interface should consists of. The specification could then be implemented in different ways, depending on device and user needs, and this implementation would be selected on deployment. We will discuss this more in the follow-up scenario “Multimedia assistant”, where user interfaces are more important.

One important challenge in composition is matching the output and input when connecting two components. The third version touched on the need to add a transformation which processes the output of one component, to provide the right input for the next. In this case Ove used this for performing calculations on data. Transformation components could also be used to extract a part of the data to use as input for a component, or change its format.

In the final version Ove added tracking of Oda's progress. This is a way of checking if transpired events match a plan. In the general case, components can send events to an event manager, and this component has been provided with a plan giving the sequence and timing of events. This is a generally useful mechanism, but challenging to put together.

Messaging is another important feature, present in all versions of the service. Messages can be put into the system as part of the service, entered by service users as in a chat, or stored in a component like the calendar. This service first sent messages using general protocols, to be received outside the scope of the service, but later switched (at least for messages to Oda) to a client included in the service.

It should be noted that many of the building blocks used here gives the composed service access to online enablers, rather than containing the functionality themselves. The calendar is present as an enabler online, and building blocks are provided for such tasks as entering an appointment, viewing appointments and getting alerts. It is also possible to use the same calendar outside the composed service; an important point.

The total result could be viewed as a collection of services that collaborate rather than a single unit. It is distributed, with different parts on different devices communicating with each other and the online enablers, though the composer needs only consider what parts to deploy where. Ove deploys a calendar interface on his own computer. Both it and the alerts starting the schedule use the same calendar, but he doesn't need to specify where the scheduled components are located. The underlying system will ensure messages are sent across a net after he deploys message receiver clients on Oda's computer and phone and connects them to the message sender component. Its distributed nature follows in part from the fact that there are two different users with different roles.

External evaluation

The scenario has been evaluated at the Norwegian Centre for Dementia Research. The following is a summary of the comments: It would be hard for Oda to use her phone for navigation. To use a map with a route is cognitively very demanding. An alternative is to have voice instructions, but many people find it embarrassing to have their phone read them instructions in public (earphones is a possible solution). The user interface is very important. When people develop cognitive impairment, it's important to keep familiar user interfaces and avoid introducing something new. Learning something new is very hard, but what has already been learned well tend to stick. When the phone gives Oda reminders, text messages must be included, as just a sound is not enough.

Related scenarios

- Multimedia assistant