Specyfikacja kalendarza wzorowanego na kalendarzu Google

# Zagadnienie tworzenia, składowania, edytowania i prezentacji serii wydarzeń.

Należy zauważyć, że nie ma przeciwwskazań (a jest wręcz wskazane), aby planowane wydarzenia przechowywać w bazie właśnie jako ich plan (definicję). Alternatywą mogłoby być przechowywanie już wygenerowanych wydarzeń na konkretny moment czasu, co z pewnością ułatwiłoby ich prezentację w oczekiwanym okresie czasu, ale byłaby to jedyna korzyść i całe mnóstwo kłopotów (jak choćby: co z wydarzeniami, którym nie określono końca). Biorąc pod uwagę, że kalendarz zawsze prezentuje wydarzenia w skończonym okresie czasu (i maksymalnie jest to okres konkretnego miesiąca), zawsze można wygenerować na podstawie istniejących definicji wydarzeń wszystkie te, które w owym okresie powinny mieć miejsce. Co więcej takie podejście pozwala za pomocą tego samego algorytmu uzyskać mające mieć miejsce wydarzenia zarówno do wspomnianej prezentacji, jak i realizacji ich obsługi (np. alerty, powiadomienia, itd.). Koszt takiego wygenerowania nie powinien być odczuwalny, należy bowiem brać pod uwagę, że z kalendarza korzysta człowiek, a ten ma jakieś 18 godzin aktywności dziennie, którą może spożytkować, zatem ilości wydarzeń w przeciągu miesiąca (czyli maksymalnego okresu czasu) są na tyle małe, że da się je wystarczająco szybko wygenerować.

Na pewno jednak sama definicja wydarzenia powinna posiadać zapisany okres obowiązywania wynikający co prawda z owej definicji, ale ze względu na prostotę pobierania powinien on być literalnie zapisany wraz z definicją.

To podejście procentuje także w przypadku edycji wydarzenia. Wystarczy bowiem dokonać zmiany w jednym wydarzeniu (ewentualnie dodatkowo je dublując - o czym za chwilę), aby przy kolejnym generowaniu danych do widoku otrzymać nowy plan wydarzeń w oczekiwanym okresie czasu (przypominam, że rozpatrujemy serie wydarzeń).

W kontekście edycji wydarzenia należy zwrócić uwagę, że ze względu na wygodę użytkownika, dobrze byłoby, aby dało się edytować dowolne wydarzenie z serii i wpływać na wszystkie pozostałe lub wręcz przeciwnie, poinformować kalendarz, że od tego wydarzenia w serii (lub wyłącznie to wydarzenie w serii) mają się zmienić reguły jego występowania. W takich przypadkach konieczne jest powielenie definicji wydarzenia. Jeżeli od wszystkie następne wydarzenia od właśnie edytowane mają się zachowywać w inny sposób, należy powielić definicję, a dotychczasową oznaczyć jako serię wydarzeń o ustalonym momencie zakończenia, który przypada na wydarzenie poprzedzające właśnie edytowane. Jeżeli rozpatrujemy przypadek, iż jedynie obecnie edytowane wydarzenie ma się zachować inaczej (np. wystąpić w innych godzinach), to należy wykonać dwie kopie dotychczasowej pojedynczej definicji wydarzenia. Jedną będącą wydarzeniem jednorazowym i kolejną, która rozpoczyna się wraz z czasem, w jakim miało wystąpić kolejne wydarzenia po właśnie edytowanym i kontynuowane jest zgodnie z pierwotną definicją jego końca. Definicja wydarzenia oryginalnego powinna zostać zmodyfikowana tak samo jak w poprzednim przypadku (czyli ustalam jej koniec na wydarzenie poprzedzające właśnie modyfikowane).

Rozpatrując prezentację wydarzeń cyklicznych (seryjnych) należy wziąć pod uwagę, że czas ich trwania może być dłuższy niż jeden dzień (dotyczy to zresztą także wydarzeń jednorazowych), dlatego należy uwypuklić iż w oczekiwanym okresie: zaczynają się i kończą, trwają, zaczynają się i trwają oraz trwają i kończą się. W kalendarzu Google fakt trwania oznacza się poprzez strzałkę z jednej lub obu stron bloku wydarzenia w ramach danego widoku. Dlatego niezbędne jest kreowanie na podstawie okresu obowiązywania wydarzenia odpowiednich danych wspomagających taką prezentację (propozycją realizacji tej potrzeby jest IntervalSlice).

Na koniec należy zauważyć, że pojedyncze wydarzenia (nie posiadające serii) wplatają się bezproblemowo w powyższy mechanizm.

## Kwestie preferencji lokalnych

Ponieważ dzień zaczynający tydzień może różnić się w zależności od kraju należy brać ten fakt pod uwagę podczas generowania wydarzeń seryjnych w okresie tygodnia. Jeśli bowiem wydarzenie ma się zacząć w środę i powtarzać przez kolejne 6 razy w tym dniu oraz w niedzielę, to w przypadku gdy pierwszym dniem tygodnia jest poniedziałek wydarzenie zakończy się w trzecim tygodniu, a jeśli to niedziela rozpoczyna tydzień, dopiero w czwartym tygodniu. Jest to istotne z punktu widzenia określenia okresu obowiązywania wydarzenia, co ma bezpośredni wpływ na jego prawidłowe prezentowanie oraz przetwarzanie.

Oczywiście najlepiej, jeśli początek tygodnia będzie (tak jak w kalendarzu Google) ustalany na poziomie konfiguracji (zaś domyślnie powinien być ustawiony zgodnie z ustawieniami kraju - czego Google akurat nie robi, {zapewne dlatego, że zabraniam ustalania mojej lokalizacji}). To samo powinno zresztą dotyczyć takich kwestii jak format daty i czasu, co nie ma już co prawda wpływu na zarządzanie kalendarzem, a jedynie na sposób jego prezentacji.

## Wspólne dane powielonej definicji

Jak można zauważyć w kalendarzu Google pomimo rozbicia definicji na dwie osobne, nadal część tej definicji jest wspólna (np. kolor). Trudno mi było dociec dlaczego tak jest to realizowane, tj. czemu są wyjątki od reguły (czemu to służy). Jeśli byłaby taka konieczność (wspólna część definicji) to wystarczyłoby wydzielić w tym celu klasę, które klasy definicji współdzieliłyby. Dopóki nie następowałaby rozbieżność pomiędzy tymi wspólnymi danymi, dopóty byłaby to jedna instancja klasy dla każdej definicji. W momencie wystąpienia rozbieżności należałoby powielić ową wspólną instancję i referencjonować już każdą definicję do odpowiedniego egzemplarza, właśnie jej dedykowanego.

# Baza danych

Jeśli chodzi o wybór danych to wybrałbym jedyną znaną mi czyli SQL Server. Dostęp z poziomu .NET zrealizowałbym za pomocą Entity Framework w podejściu Code First. W chwili obecnej wydaje mi się, że układ tabel odpowiadałby 1:1 klasom domeny. Zapewne część właściwości (przykładem może być: *Repetition.OnCertainDaysOfWeek*) nie byłaby przechowywana w bazie. Konfigurację mapowania klas do tabel zrealizowałbym za pomocą Fluent API, ponieważ mechanizm Data Annotation nie umożliwia mapowania właściwości protected, a takowe musiałyby być przechowywane w bazie (przykładem może być: *Repetition.EncodedDaysOfWeek*). Na chwilę obecną wydaje się, że LINQ to SQL wystarczyłby do odpytywania o dane. Szczególnie, że jak rozważałem wcześniej nie byłyby to zbyt duże zestawy danych (narastałyby one dopiero w wyniku generowania rzeczywistych wydarzeń), a same zapytania do bazy nie powinny być zbyt rozbudowane (prosty wybór definicji obowiązujących w przetwarzanym okresie).

# Architektura

Jeśli chodzi o architekturę rozwiązań Web, to mam skromne doświadczenie. Analizując zachowanie kalendarza Google przypuszczam, że mógłby wystarczyć wzorzec MVC wsparty o AJAX. Musiałbym sięgnąć po przykłady z projektu, w którym uczestniczyłem, aby zorientować się jak realizuje się takie pseudomodalne okno (od ustalania parametrów powtórzeń) lub poszperać w sieci. Musiałbym też podejrzeć, jak realizowane są poszczególne widoki kalendarza (zauważyłem, że np. w widoku miesięcznym dopuszczają one tylko określoną ilość wydarzeń, a potem umieszczają znacznik informujący o większej ich ilości niż widać obecnie).